



Take Command 16.03

Published By
JP Software Inc.
P.O. Box 328
Chestertown, MD 21620 USA
<http://jpsoft.com>

Table of Contents

Part I Overview	1
Part II What's New in Version 16	2
1 Version 15	13
2 Version 14	25
3 Version 13	40
4 Version 12	58
5 Version 11	73
6 Version 10	84
Part III Take Command	98
1 Starting Take Command	98
Take Command Startup Options	99
2 Take Command Interface	100
Take Command Window	101
Menus	103
Quick Options	104
Home	105
Edit	106
View	107
Tabs	108
Tools	110
Windows	110
Options	110
Help	111
Tabbed Tool Bar	112
Folders	113
List View	113
Command Input	114
Tab Windows	116
Status Bar	117
Keyboard Shortcuts	117
Context Menus	118
Using the Scrollback Buffer	119
Highlighting & Copying Text	119
Resizing the Window	120
Drag & Drop	121
Take Command Dialogs	121

Run Program	122
Find Files	122
Tab Window Toolbar.....	124
3 Configuration Options	125
Initialization (.INI) Files	126
Take Command Configuration Dialog	127
Windows.....	127
Tabs	129
Advanced.....	131
Register	133
Windows Explorer Integration	133
4 Take Command and TCC Integration	134
Part IV TCC	134
1 Comparing TCC, TCC/LE, and CMD	135
2 Starting TCC	149
TCC Startup Options	149
TCSTART and TCEXIT	153
TCC Exit Codes	155
3 Commands	155
Commands by Name	155
Commands by Category	159
Command Dialogs	165
?	165
ACTIVATE	166
ALIAS	168
ASSOC	178
ASSOCIATE	179
ATTRIB	180
BATCOMP	183
BDEBUGGER / IDE	184
BEEP	190
BREAK	191
BREAKPOINT	192
CALL	192
CANCEL	193
CD / CHDIR	194
CDD	196
CHCP	198
CLIPMONITOR.....	199
CLS	199
COLOR	200
COPY	201

DATE	209
DATEMONITOR.....	210
DEBUGMONITOR.....	211
DEBUGSTRING	212
DEFER	212
DEL / ERASE	212
DELAY	217
DESCRIBE	218
DESKTOP	221
DETACH	221
DIR	222
DIRHISTORY	234
DIRS	235
DISKMONITOR	236
DO	237
DRAWBOX	242
DRAWHLINE	243
DRAWLINE	244
ECHO	245
ECHOERR	246
ECHOS	247
ECHOSERR	248
ECHOX	248
ECHOXERR	249
EJECTMEDIA	249
ENDLOCAL	249
ESET	251
EVENTLOG	252
EVENTMONITOR.....	253
EVERYTHING	255
EXCEPT	256
EXIT	257
FFIND	258
FIREWIREMONITOR	263
FOLDERMONITOR.....	264
FONT	266
FOR	266
FREE	275
FTYPE	275
FUNCTION	276
GLOBAL	280
GOSUB	281
GOTO	283

GZIP	284
HEAD	285
HELP	288
HISTORY	288
IF	290
IFF	291
IFTP	293
INKEY	295
INPUT	297
JABBER	299
KEYBD	300
KEYS	301
KEYSTACK	301
LIST	303
LOADBTM	309
LOADMEDIA	310
LOG	310
LUA	311
MD / MKDIR	312
MEMORY	314
MKLINK	314
MKLNK	315
MOVE	317
MSGBOX	323
NETMONITOR	326
ON	327
OPTION	330
OSD	331
PATH	332
PAUSE	333
PDIR	333
PLAYAVI	338
PLAYSOUND	339
PLUGIN	340
POPD	341
POSTMSG	342
PRINT	342
PRIORITY	343
PROCESSMONITOR	343
PROMPT	344
PUSHD	346
QUERYBOX	347
QUIT	349

RD / RMDIR	349
REBOOT	351
RECYCLE	352
REM	352
REN / RENAME	353
RESOLUTION	357
RETURN	357
RExec	358
RSHell	359
SCREEN	360
SCREENMONITOR	361
SCRIPT	361
SCRPUT	362
SELECT	363
SENDHTML	367
SENDMAIL	369
SERVICEMONITOR	372
SERVICES	373
SET	373
SETARRAY	378
SETERROR	378
SETDOS	378
SETLOCAL	382
SHIFT	383
SHORTCUT	384
SHRALIAS	385
SMPP	386
SNMP	387
SNPP	387
START	388
STATUSBAR	392
SWITCH	392
SYNC	394
TAIL	398
TAR	400
TASKBAR	402
TASKDIALOG	403
TASKEND	404
TASKLIST	405
TCDIALOG	406
TCFILTER	408
TCTOOLBAR	408
TEE	410

TEXT	411
TIME	412
TIMER	413
TITLE	415
TOUCH	415
TPIPE	418
TRANSIENT	441
TREE	441
TRUENAME	443
TYPE	444
UNALIAS	446
UNFUNCTION	447
UNGZIP	448
UNSET	449
UNSETARRAY	451
UNTAR	451
UNZIP	453
USBMONITOR	455
VBEEP	456
VER	456
VERIFY	457
VIEW	457
VIEW Command Line Options	461
/Delete Option	463
Find String Option: /S	463
/Fix Option	465
Font Options	465
View ing Redirected Output	466
Printing Options	466
Text Only Printing Options	469
EBCDIC Options	470
Sending Error Reports	470
The File View	471
The Toolbar	472
Text Mode	474
Unicode Files	475
Flat Text mode	476
End of Line (EOL)	477
Tabs	479
Start Offset	479
The Status Bar	480
EBCDIC Mode	481
The Ruler	482

Line Numbers	484
Line Wrapping	485
Column Fixing	486
DOS/OEM Character Set.....	486
Display Fonts	486
Greenbar Mode.....	487
File Attributes / Line Lengths.....	488
Hex Mode	489
File Tailing	489
View ing the Clipboard Contents.....	490
Customizing Colors.....	490
Split File View	491
Synchronized Scrolling.....	493
Multiple File Window s.....	493
Arranging File Window s.....	493
Auto-Arranging File Window s.....	495
Synchronized Scrolling.....	495
Hex Mode.....	495
Hex Formats	497
Hex Line Length.....	498
Hex Font	498
Vertical Hex Mode.....	498
Searching.....	498
Search String	502
Search Options.....	503
Search Count	504
Search Skip	504
Column Search.....	504
Tips on using the Keyboard.....	505
Scrolling	506
Smooth Scrolling.....	508
IntelliMouse Support.....	508
Goto	509
Goto and Chunks.....	510
Bookmarks	511
Numbered Bookmarks.....	512
Sending selected text to your brow ser.....	512
Open Selection in V.....	512
Maintaining File Position.....	513
Paginated Files	513
The Search Bar.....	514
Continuing Search onto Next File.....	516
Block Marking / Text Highlighting.....	516

Using The Keyboard.....	519
Status Bar	519
Selecting words.....	520
Marking Columns.....	520
Adding Columns.....	521
Copying text to the clipboard.....	521
Appending to the clipboard.....	522
Copying to a file.....	522
GridLines.....	522
Creating Grids	524
Organizing Grids.....	526
Wrap Options	527
Automatically Loading Grids.....	527
Using Rules (Regular Expressions) to Load Grids.....	529
Associating with a File Extension.....	530
Exporting Grids.....	530
Importing Grids	532
Exporting Data to CSV.....	533
File Chunks.....	533
EBCDIC Files.....	535
EBCDIC File Formats.....	536
RECFM=F	537
RECFM=U	538
Carriage Control.....	539
Preferences / Configuration Options.....	539
File Options	540
More Options	542
Window Layout.....	543
EBCDIC	544
Editor/CMD	545
Fonts	547
Keyboard	548
Line Numbers	548
Search	550
Favorites.....	550
Add To Favorites.....	551
Organizing Favorites.....	552
Favorite Files	553
Sorting Favorites.....	554
Exporting/Importing Favorites.....	554
Using Numeric Drive Letters in Paths.....	555
User Commands.....	555
Organizing User Commands.....	557

User Command Options.....	558
Option Specifiers.....	559
Sorting User Commands.....	561
Exporting/Importing User Command.....	561
Using Numeric Drive Letters in Paths.....	561
Printing Files.....	562
Wrapping Lines.....	565
Print Range	566
Headers and Footers.....	567
Form Feeds	568
Margins	568
Page Length	568
2UP Printing	569
Printer Fonts	569
More Printing Commands.....	570
Text Only Printing.....	571
Raw /Binary Printing.....	572
Printer Profiles	573
Keyboard Shortcuts.....	573
Customizing the Keys.....	574
Adding/Deleteing Keys.....	575
Export/Import Keys.....	576
List of Keys	576
Copyright.....	577
VOL	577
VSCRPUT	578
WAKEONLAN	579
WEBFORM	579
WEBUPLOAD	580
WHICH	582
WINDOW	583
WMIQUERY	584
Y	585
ZIP	586
ZIPSFX	588
7UNZIP	589
7ZIP	590
4 Variables & Functions	593
Array Variables	595
System Variables	595
CDPATH	595
CMDLINE	595
COLORDIR.....	596

COMSPEC.....	596
FILECOMPLETION.....	596
HISTORYEXCLUDE.....	596
PATH	596
PATHEXT.....	596
PROMPT	597
PROMPT2.....	597
RECYCLEEXCLUDE.....	597
TEMP	597
TCMD	597
TCMDVER.....	597
TITLEPROMPT.....	598
TMP	598
TREEEXCLUDE.....	598
VARIABLEEXCLUDE.....	598
CMD.EXE Variables	598
COPYCMD.....	599
DIRCMD	599
String substitution.....	599
Variables	600
Variables by Name.....	601
Variables by Category.....	605
Command Variables.....	609
!(Variable).....	610
? variable.....	611
_ ? variable.....	611
= pseudovariable.....	611
+ pseudovariable.....	611
_4VER	612
_ACSTATUS.....	612
_ADMIN	612
_AFSWCELL.....	612
_ALT	612
_ANSI	612
_BATCH	612
_BATCHLINE.....	612
_BATCHNAME.....	613
_BATCHTYPE.....	613
_BATTERY.....	613
_BATTERYLIFE.....	613
_BATTERYPERCENT.....	613
_BDEBUGGER.....	613
_BG	613

_BOOT	613
_BUILD	613
_CAPSLOCK.....	614
_CDROMS.....	614
_CHILDPID.....	614
_CI	614
_CMDLINE.....	614
_CMDPROC.....	614
_CMDSPEC.....	614
_CO	614
_CODEPAGE.....	614
_COLUMN.....	615
_COLUMNS.....	615
_CONSOLEB.....	615
_CONSOLEPIDS.....	615
_COUNTRY.....	615
_CPU	615
_CPUUSAGE.....	615
_CTRL	615
_CWD	616
_CWDS	616
_CWP	616
_CWPS	616
_DATE	616
_DATETIME.....	616
_DAY	616
_DETACHPID.....	616
_DISK	616
_DNAME	616
_DOS	617
_DOSVER.....	617
_DOW	617
_DOWF	617
_DOWI	617
_DOY	617
_DRIVES	617
_DST	617
_DVDS	618
_ECHO	618
_EDITMODE.....	618
_ELEVATED.....	618
_EXECARRAY.....	618
_EXECSTR.....	618

_EXIT	618
_EXPANSION.....	618
_FG	618
_FTPERROR.....	618
_HDRIVES.....	619
_HLOGFILE.....	619
_HOST	619
_HOUR	619
_HWPROFILE.....	619
_IDE	619
_IDLETICKS.....	619
_IDOW	619
_IDOWF	619
_IFTP	620
_IFTPS	620
_IMONTH.....	620
_IMONTHF.....	620
_INNAME.....	620
_INSERT	620
_IP	620
_IPADAPTER.....	620
_IPADAPTERS.....	620
_IPARPPROXY.....	620
_IPDNS	620
_IPDNSSERVER.....	620
_IPROUTING.....	621
_ISFTP	621
_ISODATE.....	621
_ISODOWI.....	621
_ISOWDATE.....	621
_ISOWEEK.....	621
_ISOWYEAR.....	621
_KBHIT	621
_LALT	621
_LASTDIR.....	621
_LASTDISK.....	621
_LCTRL	621
_LOGFILE.....	622
_LSHIFT	622
_MINUTE	622
_MONITORS.....	622
_MONTH	622
_MONTHF.....	622

_NUMLOCK.....	622
_OPENAFS.....	622
_OSBUILD.....	622
_PARENT.....	622
_PID	623
_PIPE	623
_PPID	623
_RALT	623
_RCTRL	623
_READY	623
_REGISTERED.....	623
_ROW	623
_ROWS	623
_RSHIFT	623
_RUBYTYPE.....	624
_RUBYVALUE.....	624
_SCROLLLOCK.....	624
_SECOND.....	624
_SELECTED.....	624
_SERIALPORTS.....	624
_SERVICE.....	624
_SHELL	624
_SHIFT	624
_SHORTCUT.....	625
_SHRALIAS.....	625
_STARTPATH.....	625
_STARTPID.....	625
_STDIN	625
_STDOUT.....	625
_STDERR.....	625
_STZN	625
_STZO	625
_SYSERR.....	625
_TCCVER.....	625
_TCEXIT	626
_TCFILTER.....	626
_TCFOLDER.....	626
_TCLISTVIEW.....	626
_TCSTART.....	626
_TCTAB	626
_TCTABACTIVE.....	626
_TCTABS.....	626
_TIME	626

_TRANSIENT.....	626
_TZN	626
_TZO	627
_UNICODE.....	627
_UTCDATE.....	627
_UTCDATETIME.....	627
_UTCHOUR.....	627
_UTCISODATE.....	627
_UTCMINUTE.....	627
_UTCSECOND.....	627
_UTCTIME.....	627
_VERMAJOR.....	627
_VERMINOR.....	627
_VERSION.....	628
_VIRTUALBOX.....	628
_VIRTUALPC.....	628
_VMWARE.....	628
_VOLUME.....	628
_VXPIXELS.....	628
_VYPIXELS.....	628
_WINDIR	628
_WINFGWINDOW.....	628
_WINNAME.....	628
_WINSYSDIR.....	628
_WINTICKS.....	628
_WINTITLE.....	628
_WINUSER.....	628
_WINVER.....	628
_WOW64.....	629
_WOW64DIR.....	629
_X64	629
_XMOUSE.....	629
_XPIXELS.....	629
_XWINDOW.....	629
_YEAR	629
_YMOUSE.....	629
_YPIXELS.....	629
_YWINDOW.....	629
ERRORLEVEL.....	629
Functions	630
Functions by Name.....	632
Functions by Category.....	639
Date Display Formats.....	646

@ABS	647
@AFSCELL.....	647
@AFSMOUNT.....	647
@AFSPATH.....	648
@AFSSYMLINK.....	648
@AFSVOLID.....	648
@AFSVOLNAME.....	648
@AGEDATE.....	648
@ALIAS	648
@ALTNAME.....	649
@ARRAYINFO.....	649
@ASCII	649
@ASSOC.....	650
@ATTRIB.....	650
@AVERAGE.....	651
@B64DECODE.....	651
@B64ENCODE.....	651
@BALLOC.....	652
@BFREE.....	652
@BPEEK.....	652
@BPEEKSTR.....	652
@BPOKE.....	653
@BPOKESTR.....	653
@BREAD.....	654
@BWRITE.....	654
@CAPI	654
@CAPS	655
@CDROM.....	655
@CEILING.....	656
@CHAR	656
@CLIP	656
@CLIPW	657
@COLOR.....	657
@COMMA.....	658
@COMPARE.....	659
@CONSOLE.....	659
@CONSOLEB.....	659
@CONVERT.....	659
@COUNT.....	660
@CRC32.....	660
@CWD	661
@CWDS	661
@DATE	661

@DATECONV.....	662
@DAY	662
@DEBUG.....	663
@DEC	663
@DECIMAL.....	663
@DESCRIPT.....	663
@DEVICE.....	664
@DIGITS	664
@DIRSTACK.....	664
@DISKFREE.....	665
@DISKTOTAL.....	665
@DISKUSED.....	666
@DOMAIN.....	666
@DOW	666
@DOWF	667
@DOWI	667
@DOY	668
@DRIVETYPE.....	668
@DRIVETYPEPEEX.....	669
@ENUMSOURCES.....	669
@ENUMSHARES.....	670
@ERRTEXT.....	670
@EVAL	671
@EXEC	675
@EXECARRAY.....	675
@EXECSTR.....	675
@EXETYPE.....	676
@EXPAND.....	676
@EXT	677
@FIELD	677
@FIELDS.....	678
@FILEAGE.....	679
@FILEARRAY.....	679
@FILECLOSE.....	679
@FILEDATE.....	680
@FILEHANDLE.....	680
@FILENAME.....	680
@FILEOPEN.....	681
@FILEREAD.....	681
@FILEREADDB.....	682
@FILES	683
@FILESEEK.....	683
@FILESEEKLN.....	684

@FILESIZE.....	685
@FILETIME.....	686
@FILEWRITE.....	686
@FILEWRITEB.....	687
@FILTER.....	687
@FINDCLOSE.....	688
@FINDFIRST.....	688
@FINDNEXT.....	689
@FLOOR.....	689
@FOLDERS.....	690
@FONT	690
@FORMAT.....	691
@FORMATN.....	691
@FORMATNC.....	692
@FSTYPE.....	692
@FTYPE.....	693
@FULL	693
@FUNCTION.....	693
@GETDATE.....	694
@GETDATETIME.....	694
@GETDIR.....	694
@GETFILE.....	695
@GETFOLDER.....	697
@GROUP.....	698
@HEXDECODE.....	699
@HEXENCODE.....	699
@HISTORY.....	699
@HTMLDECODE.....	699
@HTMLENCODE.....	699
@IDOW	700
@IDOWF.....	700
@IF	700
@INC	701
@INDEX	701
@INIREAD.....	702
@INIWRITE.....	703
@INODE	703
@INSERT.....	704
@INSTR	704
@INT	705
@IPADDRESS.....	705
@IPADDRESSN.....	705
@IPALIASES.....	705

@IPDESC.....	706
@IPDHCP.....	706
@IPGATEWAY.....	706
@IPIPV6	706
@IPNAME.....	706
@IPNAMEN.....	707
@IPPHYSICAL.....	707
@IPPORT.....	707
@IPSERVICEALIASES.....	707
@IPSUBNET.....	707
@IPTYPE.....	707
@IPWINS.....	708
@IPWINSSERVER.....	708
@IPZONEID.....	708
@ISALNUM.....	708
@ISALPHA.....	709
@ISASCII.....	709
@ISCNTRL.....	710
@ISDIGIT.....	710
@ISFLOAT.....	710
@ISLOWER.....	711
@ISODOWI.....	711
@ISO_WEEK.....	711
@ISOWYEAR.....	711
@ISPRIME.....	711
@ISPRINT.....	712
@ISPROC.....	712
@ISPUNCT.....	712
@ISSPACE.....	713
@ISUPPER.....	713
@ISXDIGIT.....	713
@JUNCTION.....	714
@LABEL.....	714
@LCS	714
@LEFT	714
@LEN	714
@LFN	715
@LINE	715
@LINES	716
@LINKS	716
@LOWER.....	716
@LTRIM	717
@LUA	717

@MACADDRESS.....	717
@MAKEAGE.....	717
@MAKEDATE.....	718
@MAKETIME.....	718
@MAX	719
@MD5	719
@MIN	719
@MONTH.....	720
@MX	720
@NAME	720
@NUMERIC.....	721
@OPTION.....	721
@OWNER.....	722
@PATH	722
@PERL	722
@PID	722
@PIDCOMMAND.....	722
@PING	723
@PLUGINVER.....	723
@PPID	723
@PRIME	723
@PRIORITY.....	723
@PROCESSTIME.....	724
@PYTHON.....	724
@QUOTE.....	724
@RANDOM.....	724
@READSCR.....	725
@READY.....	725
@REGBREAD.....	725
@REGBWRITE.....	726
@REGCOPYKEY.....	726
@REGCREATE.....	726
@REGDELKEY.....	727
@REGEX.....	727
@REGEXINDEX.....	727
@REGEXIST.....	727
@REGEXSUB.....	727
@REGQUERY.....	728
@REGSET.....	728
@REGSETENV.....	728
@REGTYPE.....	729
@REMOTE.....	729
@REMOVABLE.....	729

@REPEAT.....	729
@REPLACE.....	730
@REREPLACE.....	730
@REVERSE.....	730
@REXX	731
@RIGHT	731
@RTRIM	731
@RUBY	731
@SCRIPT.....	732
@SEARCH.....	732
@SELECT.....	732
@SERIAL.....	733
@SERIALPORTCLOSE.....	733
@SERIALPORTFLUSH.....	733
@SERIALPORTOPEN.....	733
@SERIALPORTREAD.....	734
@SERIALPORTWRITE.....	734
@SERVER.....	734
@SERVICE.....	735
@SFN	736
@SHA1	736
@SHA256.....	736
@SHA384.....	737
@SHA512.....	737
@SHFOLDER.....	737
@SIMILAR.....	738
@SMCLOSE.....	738
@SMOPEN.....	739
@SMPEEK.....	739
@SMPOKE.....	739
@SMREAD.....	739
@SMWRITE.....	740
@SNAPSHOT.....	740
@STRIP	740
@SUBSTR.....	740
@SUBST.....	741
@SUMMARY.....	741
@SYMLINK.....	742
@SYSTEMTIME.....	742
@TALNUM.....	742
@TALPHA.....	742
@TARCOUNT.....	743
@TARCFILE.....	743

@TARFILE.....	743
@TARFILEDATE.....	743
@TARFILESIZE.....	743
@TASCII.....	743
@TCL	744
@TCNTRL.....	744
@TDIGIT	744
@TIME	744
@TIMER.....	745
@TK	745
@TLOWER.....	745
@TPRINT.....	746
@TPUNCT.....	746
@TRIM	746
@TRIMALL.....	746
@TRUENAME.....	747
@TRUNCATE.....	747
@TSPACE.....	747
@TUPPER.....	748
@TXDIGIT.....	748
@UNC	748
@UNICODE.....	748
@UNIQUE.....	749
@UNQUOTE.....	749
@UNQUOTES.....	749
@UPPER.....	750
@URLDECODE.....	750
@URLENCODE.....	750
@UTF8DECODE.....	750
@UTF8ENCODE.....	750
@UDECODE.....	750
@UUENCODE.....	750
@UUID	750
@VERINFO.....	751
@VERSION.....	751
@WATTRIB.....	752
@WILD	752
@WINAPI.....	753
@WINCLASS.....	754
@WINCLIENTSIZE.....	754
@WINEXENAME.....	754
@WININFO.....	754
@WINMEMORY.....	754

@WINMETRICS.....	755
@WINPID.....	756
@WINPOS.....	756
@WSIZE.....	757
@WSTATE.....	757
@WNSYSTEM.....	757
@WTITLE.....	759
@WMI	759
@WORD.....	760
@WORDS.....	760
@WORKGROUP.....	761
@XMLCLOSE.....	761
@XMLNODES.....	761
@XMLOPEN.....	761
@XMLXPATH.....	761
@YEAR	762
@YDECODE.....	762
@YENCODE.....	762
@ZIPCFFILE.....	762
@ZIPCFFILESIZE.....	762
@ZIPCOMMENT.....	762
@ZIPCOUNT.....	763
@ZIPDFILE.....	763
@ZIPFILECRC.....	763
@ZIPDFILESIZE.....	763
@ZIPFILECOMMENT.....	763
@ZIPFILEDATE.....	763
5 Command Line	763
Command Line Editing	764
Command History and Recall	766
Command History Window	768
Local and Global History Lists	769
Command Names & Parameters	770
Conditional Expressions	770
Filename Completion	775
Customizing Filename Completion	777
Filename Completion Window	778
Converting Between Long & Short Filenames	778
Appending Backslashes to Directory Names	779
Extended Parent Directory Names	779
Directory History Window	780
Variable Name Completion	781
Expanding and Disabling Aliases	782

Multiple Commands	782
Conditional Commands	782
Command Grouping	783
Starting Applications	785
Waiting for Applications to Finish	786
Escape Character	786
Command Parsing	787
Command Line Length Limits	789
Special Character Compatibility	789
Date Input Formats	790
Case Sensitivity	790
Directory Navigation	790
CDPATH	791
Extended Directory Searches.....	792
Automatic Directory Changes.....	795
Directory Aliases.....	796
6 Aliases & Batch Files	796
Aliases	797
Batch Files	799
.BAT, .CMD & .BTM Files.....	800
Echoing in Batch Files.....	800
Special syntax for CMD compatibility.....	800
Batch File Line Continuation.....	801
Batch File Parameters.....	801
Parameter Quoting.....	802
Using Environment Variables.....	804
Batch File Commands.....	804
Interrupting a Batch File.....	805
Detecting TCC and Take Command.....	806
Using Aliases in Batch Files.....	806
Debugging Batch Files.....	808
String Processing.....	808
Batch File Compression.....	810
Lua support.....	810
Perl support.....	811
Python support.....	811
REXX Support.....	811
Ruby support.....	812
Tcl/tk Support.....	812
EXTPROC / SHEBANG Support.....	812
7 File Selection	813
Wildcards and Regular Expressions	813
Executable Extensions	817

Using Internet URLs	818
Using FTP and HTTP Servers	818
OpenAFS	823
Ranges	823
Size Ranges.....	825
Date Ranges.....	826
Time Ranges.....	828
File Exclusion.....	829
Owner Ranges.....	830
Description Ranges.....	830
Attribute Switches	830
Multiple Filenames	832
Include Lists	833
Delayed Variable Expansion	834
Extended Parent Directory Names	834
LFN File Searches	834
@File Lists	835
Switches for File Selection	836
8 Input / Output Redirection	837
Redirection and Pipes	837
Redirection.....	838
Pipes	841
ANSI X3.64 Support	843
Keystack	843
Page and File Prompts	844
9 Configuration	844
Initialization Files	845
Directives.....	846
Key Mapping Directives.....	848
General Input Keys.....	849
Backspace	850
BeginLine	850
Copy (directive).....	850
Del (directive)	850
DelToBeginning	850
DelToEnd	850
DelWordLeft	851
DelWordRight	851
Down	851
EndLine	851
EraseLine	851
ExecLine	851
Ins	852

Left	852
NormalKey	852
Paste	852
Right	852
Up	853
WordLeft	853
WordRight	853
Command Line Editing Keys	853
AddFile	854
AliasExpand	854
CommandEscape	854
DellHistory	854
DirWinOpen	854
EndHistory	854
Help (directive)	855
HelpWord	855
HistWinOpen	855
LastHistory	855
LFNToggle	855
LineToEnd	855
NextFile	856
NextHistory	856
NormalEditKey	856
PopFile	856
PrevArgument	856
PrevFile	856
PrevHistory	857
Redo	857
RepeatFile	857
SaveHistory	857
Undo	857
VariableExpand	857
LIST Keys	858
ListBack	858
ListClipboard	858
ListContinue	858
ListExit	858
ListFind	859
ListFindRegex	859
ListHex	859
ListHighBit	859
ListInfo	859
ListNext	859

ListOpen	859
ListPrevious	860
ListPrint	860
ListRefresh	860
ListUnicode	860
ListWrap	860
NormalListKey	860
Popup Window Keys	860
NormalPopupKey	861
PopupWinDel	861
PopupWinEdit	861
PopupWinEditWin	861
PopupWinExec	861
Advanced Directives	861
AliasSize	862
AutoFire all	862
AutoProxy	862
ClearKeyMap	862
CMDBatchDelimiters	863
CMDVariables	863
CompleteAllFiles	863
ConsolePopupWindows	863
Copyright	863
Debug	864
DelWipePasses	864
EverythingSearch	864
FileCompletionLooping	864
FilesCaseSensitive	864
FunctionSize	864
INISearch	865
LanguageDLL	865
Lua	865
MSAAMenu	865
NoINIErrors	865
StartTabWait	866
TrayHotKey	866
UpdateINI	866
UTF8Output	866
Configuration Dialog	866
Startup	867
Windows	870
Command Line	871
Advanced	874

Internet	877
Updates	880
Part V IDE / Batch Debugger	880
Part VI Troubleshooting	886
1 Troubleshooting, Service & Support	886
Technical Support	886
Contacting JP Software	888
2 Supported Platforms	889
3 Help File	889
4 Error Messages	890
Part VII Registration	896
Part VIII Reference	897
1 Updater	897
2 Windows x64	897
3 Plugins	898
4 CMD.EXE Comparison	902
5 CMD Compatibility	907
6 Limits	908
7 File Systems & File Names	909
Drives & Volumes	909
File Systems	910
Directories & Subdirectories	911
File Names	912
File Attributes	913
File Time Stamps	914
NTFS File Streams	915
8 Regular Expression Syntax	915
9 XML in TCC	922
10 Miscellaneous	924
Executable Files & File Searches	924
Windows File Associations	926
Popup Windows	926
Windows System Errors	927
ASCII Codes and Key Names	927
ASCII Tables	928
Key Names	932
ANSI X3.64 Command Reference	933
Colors, Color Names & Codes	935

Part IX Copyright & Version	937
Index	938

1 Overview



TAKE COMMAND 16.03

Welcome to our help! We have designed this help file to accompany our products **Take Command** and **TCC/LE**. (For a description of **TCC/LE**, see [TCC and TCC/LE](#)).

Take Command is designed for Windows Vista, Server 2008, Windows 7, Windows 8, and Server 2012, and is available in 32-bit and 64-bit versions. **Take Command** combines the best features of the GUI and character-mode interfaces. You can have multiple console applications open in tabbed windows, with a Windows Explorer-like interface available for those times when you need a visual look at your folders.

Take Command is composed of three elements which work closely together:

Take Command Environment - A rich development and operations environment that allows you to:

- Run multiple console and GUI applications simultaneously in tabbed windows, including our own Take Command Console (**TCC**), CMD, PowerShell and bash. **Take Command** will display output much faster (up to 10x!) than running the application in a standard Windows console window.
- Cut and paste text
- Drag and drop files into tab windows from an Explorer-like environment, other applications, or the desktop
- Create and edit command scripts with a full featured editor, including syntax highlighting
- Debug batch scripts with a sophisticated debugger, including single-stepping and conditional breakpoints

Take Command Console (TCC) - A command processor compatible with CMD (the default command processor in Windows Vista / 2008 / 7 / 8 / 2012) but substantially enhanced with thousands of additional features. **TCC** provides the ability to:

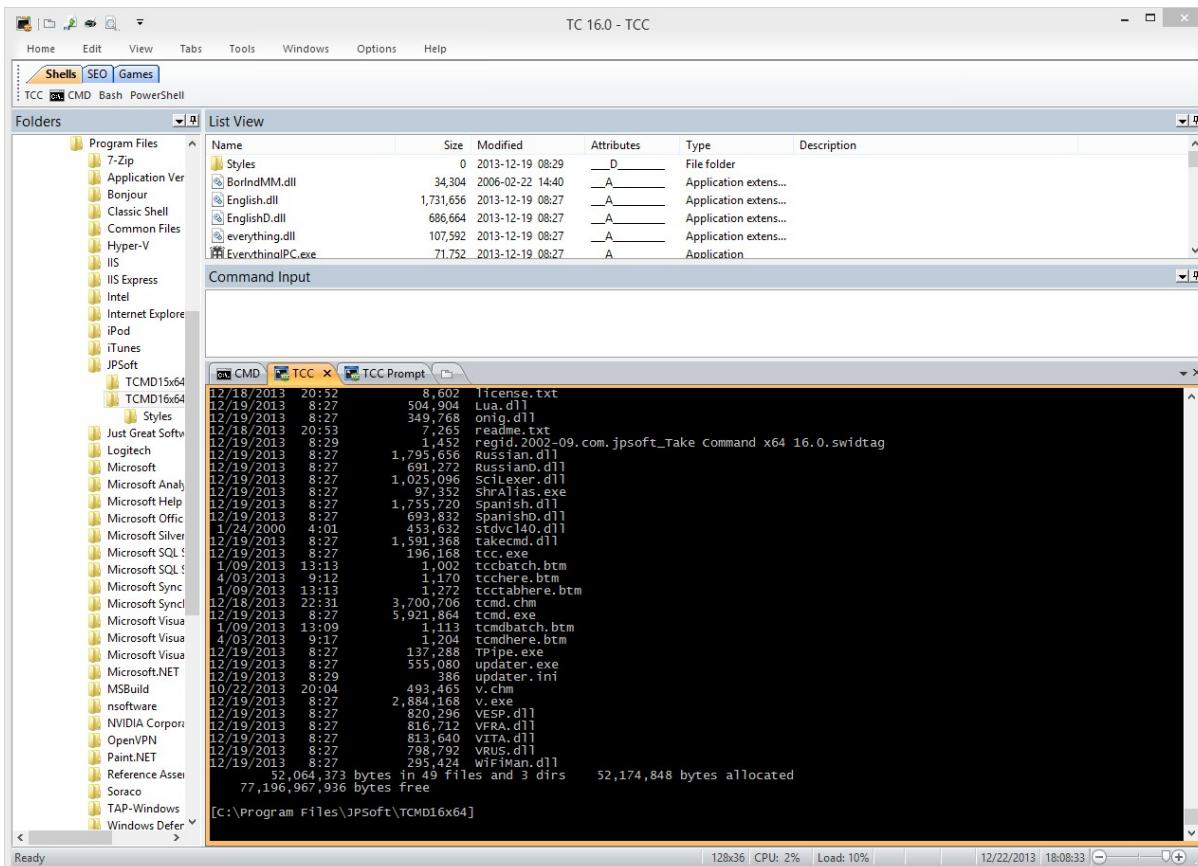
- Interactively run commands, such as DIR, COPY, etc
- Interactively run batch script files, such as .CMD, .BAT or .BTM scripts
- Run batch scripts as background processes based on timed schedules or operational triggers, such as changes in the system environment

Take Command Language - A mature scripting language based on and compatible with CMD, but massively enhanced. It includes:

- 190+ internal commands
- 330+ functions
- 250+ variables
- Hundreds of additional options for CMD compatible commands
- Additional underlying capabilities, such as the ability to access FTP and HTTP sites as if they

were local disk drives

The following image shows how the pieces fit together. The overall environment surrounds a set of consoles, each with its own tabbed window. Each console can run commands in the Take Command language (or other languages, such as PowerShell or bash, in additional tab windows).



JP Software Inc.
P.O. Box 328, Chestertown, MD 21620, USA
phone: (800) 595-8197
web: <http://jpsoft.com/>

2 What's New in Version 16

Feature List:

The Take Command UI has been revamped, to make things easier to find and to provide a little more room for the tab windows. The Take Command menu has been changed to a ribbon format (which can be optionally always displayed or minimized).

Take Command now has a customizable Quick Options toolbar that is on the left side of the caption bar. The Quick Options toolbar has a customize button on the right side that adds or removes commands from Quick Options, enables / disables the Tabbed Toolbar, and minimizes or displays

the ribbon.

Because of the change to the ribbon menu, the old styles have been deleted and new ones added to the Options menu:

Office 2007 Blue
Office 2007 Silver
Office 2007 Black
Office 2007 Aqua
Office 2007 System
Office 2010 Blue
Office 2010 Silver
Office 2010 Black
Office 2013
Windows 7
Windows 8

The Take Command Edit menu now includes Undo / Redo options (enabled if TCC is running in the active tab window).

The Command Input window context menu now includes Undo / Redo options.

The TCC tab window context menu now includes Undo / Redo options.

The IDE alias window context menu now includes Undo / Redo options.

The IDE function window context menu now includes Undo / Redo options.

The IDE environment window context menu now includes Undo / Redo options.

The IDE user-defined functions window context menu now includes Undo / Redo options.

The IDE batch parameters window context menu now includes Undo / Redo options.

The Take Command Options menu has a new Font option to set the tab window font size (system, normal, large, and extra large).

Take Command has a new menu option (Tabs / Close Other Tabs) that will close all of the tabs except the current one.

The Take Command Explorer toolbar has been removed in order to increase the default tab window size, and most of the options moved to the View tab.

The Take Command tabbed toolbars now support dragging & dropping commands from the Customize dialog onto the tabbed toolbar.

Take Command has a new menu option (View / Full Screen) that will maximize the tab windows and hide the menu & toolbar options.

There is a new Take Command menu option "Tools" that displays icons to run VIEW or the batch editor / debugger.

The Take Command "Windows" menu option now includes additional splitter options to create a new

horizontal or vertical tab group with the current tab.

There is a new small tab on the right of the tab window headers that opens a new default tab window.

Hold down the Alt key while spinning the mouse wheel to select tab windows.

If you manually detach a tab window, Take Command will no longer try to automatically reattach it if you have AutoAttachConsoles=Yes in your TCMD.INI.

Take Command now sets the hidden console windows to use a Unicode font (Consolas or Lucida Console) to avoid problems with mismatched TCMD and TCC fonts.

Take Command has a new menu option (Tabs / Send Input) that will send all input for that tab window to selected other tab windows.

Added an internal Lua interpreter. Lua scripts can be executed by entering the script name at the TCC prompt, or with the LUA command (see below). You can also execute Lua expressions with the @LUA variable function.

TCC command history now supports wildcards. For example, you can search for a previous command that contains the string "foo" by typing "*foo*" on the command line and pressing the up or down keys.

The TCC command history popup window now supports wildcards. For example, you can display all of the commands that contain the string "foo" by typing "*foo*" on the command line and pressing the PgUp key.

The TCC directory history popup window now supports wildcards. For example, you can display all of the directories that contain the string "foo" by typing "*foo*" on the command line and pressing the Ctrl-PgUp key.

The TCC popup windows can now optionally use character-mode windows instead of GUI windows. This is intended for use with server consoles that are character-mode only, or when using SSH with no GUI support; there is no benefit (and several disadvantages) in using this option for normal non-server environments. See the TCMD.INI ConsolePopupWindows directive.

The TCC popup windows now support Unicode characters in the Search edit control.

TCC customized filename completion will exclude extensions that are prefixed with a !.

New version of the GUI framework.

New versions of all of the IPWorks dll's.

New version of the TPIPE engine dll.

New version (3.3.7) of the Scintilla dll (for the IDE editor/debugger).

New version of V.EXE (for the internal VIEW command).

Batch Debugger / IDE:

If you hover the mouse over an internal or environment variable name, the IDE will show a tooltip with the current value (truncated to 512 characters if necessary).

If you hover the mouse over an internal command name, the IDE will show a tooltip with the command syntax.

INI Directives:

AliasSize=n - Increased the maximum global alias size to 512K characters.

ConsolePopupWindows=yes|NO - enable or disable character-mode popup windows (for example, command or directory history windows). This is intended for use with server consoles that are character-mode only, or when using SSH with no GUI support; there is no benefit (and several disadvantages) in using this option for normal non-server environments.

DebuggerToolTips=YES|no - enable or disable tooltips in the IDE / batch debugger that show the current value of internal and environment variables, and the syntax for internal commands.

FileCompletionLooping=yes|NO - enable or disable Linux-style filename completion looping. I.e., when TCC reaches the last match, it will loop back to the first match (with no indication that it has done so). It's easier to use the tab / F8 forward/back stepping in TCC, but some hard-wired Linux users wanted this.

FunctionSize=n - Increased the maximum global function list size to 256K characters.

Lua=YES|No - enable or disable executing Lua scripts directly from the TCC prompt.

New Internal Variables:

_IPADAPTER - returns the index of the current adapter.

_IPADAPTERS - returns the number of adapters in the system.

_IPARPPROXY - returns 1 if the local computer is acting as an ARP proxy.

_IPDNS - returns 1 if DNS is enabled for the local computer.

_IPDNSSERVER - returns the default DNS server for the local computer.

_IPROUTING - returns 1 if routing is enabled on the local computer.

_ISFTP - returns 1 if you have an SSH IFTP connection open

_7UNZIP_FILES - returns the number of files extracted in the last 7UNZIP command.

_7UNZIP_ERRORS - returns the number of errors in the last 7UNZIP command.

_7ZIP_FILES - returns the number of files compressed in the last 7ZIP command.

_7ZIP_ERRORS - returns the number of errors in the last 7ZIP command.

Updated Variable Functions:

@PPID[=pid] - if the argument begins with a =, it is assumed to be a PID instead of a process name.

@SNAPSHOT[=pid] - if the argument begins with a =, it is assumed to be a PID instead of a window title.

@WINCLIENTSIZE[=pid] - if the argument begins with a =, it is assumed to be a PID instead of a window title.

@WINEXENAME[=pid] - if the argument begins with a =, it is assumed to be a PID instead of a window title.

@WINPOS[=pid] - if the argument begins with a =, it is assumed to be a PID instead of a window title.

@WINSIZE[=pid] - if the argument begins with a =, it is assumed to be a PID instead of a window title.

@WINSTATE[=pid] - if the argument begins with a =, it is assumed to be a PID instead of a window title.

New Variable Functions:

@FONT[n] - returns console font information. *n* is the info requested:

- 0 - font name (Windows usually returns an empty string unless you've previously set the font)
- 1 - font width
- 2 - font height
- 3 - font weight
- 4 - font family
- 5 - font index in console font table

@IPADDRESSN[n] - returns the IP address of the adapter at index *n*.

@IPALIASES[name] - returns the other names corresponding to the host with the specified name.

@IPDESC[n] - returns the description for the adapter at index *n*.

@IPDHCP[n] - returns the DHCP server for the adapter at index *n*.

@IPGATEWAY[n] - returns the gateway for the adapter at index *n*.

@IPIPV6N[n] - returns the IPv6 address of the adapter at index *n*.

@IPNAMEN[n] - returns the name of the adapter at index *n*.

@IPPHYSICAL[n] - returns the physical address of the adapter at index *n*.

@IPPORT[service] - returns the port number for the specified service.

@IPSERVICEALIASES[service] - returns aliases for the specified service.

@IPSUBNET[n] - returns the subnet of the adapter at index *n*.

@IPTYPE[n] - returns the type of adapter at index *n*. Possible values include:

OTHER
WIRELESS
ETHERNET
TOKENRING
FDDI
PPP
LOOPBACK
SLIP

@IPWINS[n] - returns 1 if the adapter at index *n* uses WINS.

@IPWINSSERVER[n] - returns the primary WINS Server for the adapter at index *n*.

@IPZONEID[n] - returns the IPv6 Zone ID (also known as a scope ID) for the adapter at index *n*. The values of the Zone ID are defined relative to the sending host.

@LUA[expression] - execute a Lua expression.

@UUID[n] - creates a UUID (same as a GUID in Windows). *n* can be:

0 - returns the UUID with lower case alphabetic characters and embedded hyphens
1 - returns the UUID with upper case alphabetic characters and embedded hyphens
2 - returns the UUID with lower case alphabetic characters and no hyphens
3 - returns the UUID with upper case alphabetic characters and no hyphens

@VERSION[filename[,separator[,start[,force]]]] - return a serially "versioned" replacement for the file name if the file already exists. This is distinct from the function of @UNIQUE[] in that it retains the entire filename and only appends a version separator character and an ascending version number to the filename. @VERSION does not create the file; it just returns the next available version name.
@VERSION has four arguments:

- 1) the filename to "versionize" (required)
- 2) the version separation character (optional, defaults to ':')
- 3) the starting version number (if necessary to add a version number; optional, defaults to '1')
- 4) flag to force versioning, even if the file doesn't exist (optional, defaults to 0 or FALSE).

@WINTITLE[pid] - return the window title of the process with the specified PID.

Updated Commands:

ACTIVATE

If the window title argument begins with a =, it is assumed to be a PID instead of a title. (Note that this is less reliable than providing a title, as a process can have multiple top-level windows.)

ALIAS

/Z - Overwrite the alias list with the contents of the specified file (must be used with /R). ALIAS /R /Z is 20x faster than an ALIAS /R.

DIR

/nm:x - Display a maximum of 'x' directory entries.

DO

/Q - like /L, but treats double quoted arguments (with embedded whitespace) as a single argument.

ESET

ESET now supports filename completion when editing aliases.

FUNCTION

/Z - Overwrite the alias list with the contents of the specified file (must be used with /R). FUNCTION /R /Z is 20x faster than FUNCTION /R.

PDIR

/nm:x - Display a maximum of 'x' directory entries.

POSTMSG

If the window title argument begins with a =, it is assumed to be a PID instead of a title. (Note that this is less reliable than providing a title, as a process can have multiple top-level windows.)

TASKLIST

If the process name / window title argument begins with a =, it is assumed to be a PID instead of a name or title.

TPIPE

Unicode characters in the search / replace fields are now converted to UTF-8 before being processed by the Regular Expression engine.

Added log entries for size/date of ignored files.

Search/replace lists can now generate log entries (useful for debugging). Logs can optionally be output only for where replacements occurred.

Search/Replace lists now discard blank search terms and terms where the replacement is identical to the search.

Log filename now has environment variables resolved before display.

Named subexpressions created by a Split on Pattern filter are now saved as global variables for use in other filters.

Upgraded PDF component now handles more PDF document types.

/CLIPBOARD - Runs the current filter with input from and output to the clipboard.

/DUP - added an optional final argument to specify how the output should be formatted for Type=1 -- e.g., "%d %s" to show the count followed by the string.

/dup=Type,MatchCase,StartColumn,Length,IncludeOne,Format

/INPUTBINARY=n - Determines how binary files are processed. The options are:

- 0 - Binary files are processed (default)
- 1 - Binary files are skipped
- 2 - Binary files are confirmed before processing

/INPUTDELETE=n - If 1, the input files will be deleted after processing. USE WITH CAUTION!!

/INPUTPROMPT=n - If 1, TPIPE will prompt before processing each input files.

/OUTPUTCHANGED=n

- 0 - Always output
- 1 - Only output modified files
- 2 - Delete original if modified

/OUTPUTMODE=n - Sets the output mode. The options are:

- 0 - Output to clipboard (all files are merged)
- 1 - Output to files
- 2 - Output to a single merged file

/OUTPUTOPEN=n - If 1, TPIPE will open each output file in its associated program upon completion.

/LINE - added optional final argument (0 or 1) to not reset the line count at the end of the file.

/line=StartNumber,Increment,SkipBlank,DontNumberBlank,NumberFormat,DontReset

/OUTPUTFOLDER=folder - Set the output filter folder.

/SIMPLE - added new types:

- 81 - Shred file
- 82 - Unicode to escaped ASCII
- 83 - Restrict to Unicode files
- 84 - T-filter (process the same input in multiple ways)
- 85 - Convert HTML/XML entities to text

/SORT=Type,Reverse,RemoveDuplicates,StartColumn,Length - Sort files. Note that /SORT is slow on large files; it's intended for simple sorts of relatively small files. For big files, you should use a dedicated sorting app.

Type - the sort type

0 - ANSI sort
 1 - ANSI sort (case sensitive)
 2 - ASCII sort
 3 - ASCII sort (case sensitive)
 4 - Numeric sort
 5 - Sort by length of line
 Reverse - If 1, sort in descending order; if 0, sort in ascending order
 RemoveDuplicates - If 1, remove duplicate lines; if 0 keep duplicate lines
 StartColumn - The column in the line to begin the comparisons
 Length - The length of the comparison

/STRING - added new types:

- 17 - Restrict to filenames matching the Perl pattern
- 18 - Restrict to filenames not matching the Perl pattern

VIEW

New version of V.EXE.

New octal word hex format

Better recognition of UTF-8 files.

ZIP

ZIP is now 500% faster when zipping files.

New Commands:

7UNZIP - Unzip files in .7z archives. The syntax is similar to the UNZIP command:

```
7UNZIP [/A:[[-][+]]rhsdaecjot] /C /CRC /D /E /F /Nt /P /O /Q /S"password" /TEST /U /V]
ziparchive path file ...
```

ziparchive	The 7Zip file to work with
path	The path where files will be extracted
file	The file(s) to extract
/=	Display the 7UNZIP command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.
/A:...	Select only those files that have the specified attribute(s) set. See Attribute Switches for information on the attributes which can follow /A:. Do not use /A: with @file lists . See @file lists for details. You can specify /A:= to display a dialog to help you set individual attributes. (Not available in TCC/LE .)
/C	Display (on standard output) the contents of a file in the zip archive.
/CRC	Display the file CRCs (must be used with /V).
/D	Recreate the directory structure saved in the 7zip file.
/E	Extract the specified file(s). (This is the default.)
/F	Extract only those files that currently exist in the target folder, and which are older than the file in the 7zip archive.

/Nt	Don't update the CD / CDD extended directory search database (JPSTREE.IDX) .
/O	Overwrite existing files. 7UNZIP normally prompts before overwriting an existing file; /O will suppress the prompt.
/P	Display the progress (0 - 100%) for each file as it is extracted.
/Q	Don't display filenames as they are extracted.
/S	Use the specified password to extract the file(s) from an encrypted archive. If you don't provide a password on the command line, 7UNZIP will prompt you to enter one.
/TEST	Test the integrity of the 7zip file (header and contents). Any errors will be displayed on STDERR.
/U	Extract files which either don't exist in the target folder, or which are older than the file in the 7zip archive.
/V	View the list of files in the archive (date, time, size, and filename). If the 7zip file is password protected, 7UNZIP will append a * after the filename.

7ZIP - Zip files in .7z archives. The syntax is similar to the ZIP command:

```
7ZIP [/A:[[-][+]rhsdaecjot] /A /C /CRC /D /F /Kn /Ln /M /O:[-]adegrstu /P /Q /R /S"password" /T  
/TEST /U /V] ziparchive [@file] file...
```

ziparchive The 7zip file to work with

file The files(s) to be added to the 7zip file

/= Display the 7ZIP command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.

/A:.... Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with @file lists. See [@file lists](#) for details. You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

/A Add the specified file(s) to the 7zip file. (This is the default.)

/C Display (on standard output) the contents of a file in the 7 zip archive.

/CRC Display the file CRCs (must be used with /V).

/D Delete the specified file(s) from the 7zip file.

/Kn Compression method:

- 0 LZMA (default)
- 1 BZip2
- 2 Delta
- 3 Copy (no compression)
- 4 Deflate
- 5 LZMA2

/F Update only those files that currently exist in the 7zip file, and which are older than the files on disk.

/Ln Set the compression level (1 - 5, where 1=minimum compression, and 5=maximum compression). The default is 3.

/M Delete the files from the disk after adding them to the 7zip file.

/O:.... Sort the files before processing. (Not available in **TCC/LE**.) You may use any combination of the sorting options below. If multiple options are used, the files will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n Sort by filename and extension, unless e is explicitly included. This

	is the default.
-	Reverse the sort order for the next sort key
a	Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
d	Sort by date and time (oldest first); also see /T:acw
e	Sort by extension
g	Group subdirectories first, then files
r	Reverse the sort order for all options
s	Sort by size
t	Same as d
u	Unsorted
/P	Display the progress (0 - 100%) for each file as it is zipped.
/Q	Don't display the files being zipped.
/R	If the argument is a subdirectory, copy all of the files in that subdirectory and all of its subdirectories to the 7zip file.
/S	Use the specified password to encrypt the file(s). If you don't provide a password on the command line, 7ZIP will prompt you to enter one.
/T	Save the file attributes (they will be set when the file is extracted).
/TEST	Test the integrity of the 7ZIP file (header and contents). Any errors will be displayed on STDERR.
/U	Update files which either don't exist in the 7zip file, or which are older than the files on disk.
/V	View the list of files in the 7zip file (date, time, size, and filename). If the zip file is password protected, 7ZIP will append a * after the filename.

LUA - Invoke the internal Lua interpreter. The syntax is:

`lua [options] [script [args]]`

The options are:

- e _stat_: executes string stat;
- l _mod_: "requires" mod;
- i: enters interactive mode after running script;
- v: prints version information;
- : stops handling options;
- : executes stdin as a file and stops handling options.

After handling its options, lua runs the given script, passing to it the given args as string arguments. When called without arguments, lua behaves as `lua -v -i` when the standard input (stdin) is the console, and as `lua -` otherwise.

Before running any argument, the interpreter checks for an environment variable `LUA_INIT`. If its format is `@_filename_`, then lua executes the file. Otherwise, lua executes the string itself.

All options are handled in order, except -i. For instance, an invocation like

```
lua -e'a=1' -e 'print(a)' script.lua
```

will first set a to 1, then print the value of a (which is '=1='), and finally run the file `script.lua` with no arguments.

Before starting to run the script, lua collects all arguments in the command line in a global table called arg. The script name is stored at index 0, the first argument after the script name goes to index 1, and so on. Any arguments before the script name (that is, the interpreter name plus the options) go to negative indices. For instance, in the call

```
lua -la b.lua t1 t2
```

the interpreter first runs the file a.lua, then creates a table

```
arg = {  
    [-2] = "lua",  
    [-1] = "-la",  
    [0] = "b.lua",  
    [1] = "t1", [2] = "t2" }
```

and finally runs the file b.lua. The script is called with arg[1], arg[2], ... as arguments; it can also access these arguments with the vararg expression '`=...=`'.

In interactive mode, if you write an incomplete statement, the interpreter waits for its completion by issuing a different prompt.

If the global variable `_PROMPT` contains a string, then its value is used as the prompt. Similarly, if the global variable `_PROMPT2` contains a string, its value is used as the secondary prompt (issued during incomplete statements). Therefore, both prompts can be changed directly on the command line. For instance,

```
lua -e"_PROMPT='myprompt> '" -i
```

(the outer pair of quotes is for the shell, the inner pair is for Lua), or in any Lua programs by assigning to `_PROMPT`. Note the use of `-i` to enter interactive mode; otherwise, the program would just end silently right after the assignment to `_PROMPT`.

`WAKEONLAN` - send a "Wake-On-LAN" packet to the specified system (which may also be a broadcast address). This will power on the remote machine if the functionality is supported by the network card on the remote machine.) The syntax is:

`WAKEONLAN` `remotehost` `macaddress`

`remotehost` - the address of the machine to wake
`macaddress` - the physical address of the remote host

2.1 Version 15

Feature List:

Updated the TextPipe engine to 9.3.1.

Updated the Scintilla editor (used by the IDE and Command Input window) to 3.2.5.

Updated the installer.

Updated the registration & licensing modules. Registration can now only be done from within Take Command (Help menu), not TCC.

Take Command, TCC, and the IDE (batch debugger) have been optimized to load faster.

The Take Command help is now also available in ePUB format at <http://jpsoft.com/downloads/v15/TakeCommand.epub>. You can choose either the PDF or ePUB format for reading on your portable devices.

TCC is now supported in the Windows PE environment. (There are a few commands that won't work because of missing Windows APIs, and Take Command won't work because there is no GUI.)

Take Command now doesn't update the Folder & List Views at startup if they are disabled or set to AutoHide. (This will speed up the load time substantially if you have network drives which are mapped but unavailable.)

The password fields in TCMD.INI are now encrypted when they are saved. (The encryption is strong, but if somebody wants to debug TCC.EXE and monitor the API calls, they'll eventually be able to figure out the unencrypted strings. But they'll have to work for it.)

The Command Input window now uses the same font and point size as the tab windows.

Take Command now disables updates when renaming folders in the Folder View.

Take Command now disables updates when renaming files or directories in the List View.

Take Command now disables updates when editing descriptions in the List View.

The Take Command Folder View now supports Ctrl-C or Ctrl-Insert to copy the current selection to the clipboard.

The Take Command List View now supports Ctrl-C or Ctrl-Insert to copy the current selection(s) to the clipboard.

Take Command now supports copying descriptions in DESCRIPT.ION when copying / dragging / dropping files in the Folder & List View windows.

Added a global hotkey (default Ctrl-Shift-Z) to toggle Take Command to and from the system tray.

Updated the Internet support dll's for TCC.

Updated the zip / tar support dll's for TCC.

Added support for the new OpenAFS 1.7.x redirector when retrieving the volume information (for example, in FREE, %@DISKFREE, etc.).

The TCC command line editor has Undo and Redo support. You can remap the keys with the "Undo" and "Redo" key mapping entries in TCMD.INI.

Undo - Alt-Z

Redo - Alt-Y

Batch Editor / Debugger:

When a file has been modified, the tab title will be updated with a leading *. When the file is saved, the * will be removed.

INI Directives:

AutoProxy=YES no	Enable / disable automatic HTTP proxy detection
AutoFirewall=YES no	Enable / disable automatic firewall detection
TrayHotKey=Z	The hotkey to toggle Take Command to and from the system tray. The specified alphabetic key is combined with Ctrl + Shift, so the default hotkey is Ctrl-Shift-Z.
Copyright=YES no	Display the TCC copyright message at startup. This is the same as the TCC /Q startup option, and only applies to registered copies.
EverythingSearch=yes NO	If YES, CDD will use "Everything Search" (http://www.voidtools.com) instead of JPSTREE.IDX for fuzzy directory matching. See CDD for details.
FilesCaseSensitive=yes NO	If YES, filename comparisons will be case sensitive (like Linux, and unlike Windows).
Redo=Alt-Y	Key mapping directive to redo last edit (see Undo and Redo above).
Undo=Alt-Z	Key mapping directive to undo last edit (see Undo and Redo above).

The password fields in TCMD.INI for the Internet settings are encrypted.

Internal Variables:

%_do_loop - Incremented each time through a DO loop.

%_tclistview - Returns the selected items in the List View window as an include list.

%_virtualbox - Returns 1 if TCC is running in a VirtualBox VM.

Variable Functions:

%@DISKFREE - Now supports the OpenAFS 1.7.x redirector to retrieve disk space usage.

%@DISKTOTAL - Now supports the OpenAFS 1.7.x redirector to retrieve disk space usage.

%@DISKUSED - Now supports the OpenAFS 1.7.x redirector to retrieve disk space usage.

%@FORMAT - If the second argument (**string**) doesn't exist, @FORMAT now treats it as an empty string and pads the output accordingly.

%@LINES - Now also sets two variables:

_LINES_MAXLEN - The length of the longest line

_LINES_MAXLOC - The line number (base 0) of the longest line.

%@MACADDRESS - Returns the MAC address of the network interface at the specified address.

%@SELECT - Added optional start line and key mask fields. The start line will highlight the specified line number (the first line is 1).

The selected line number will be returned in the SELECT_LINE environment variable (the first line is 1).

If you specify a key mask, the searching is disabled, and TCC will check input keystrokes for a match against the key mask. If a match is found, @SELECT will return the current line and set the _SELECT_KEY environment variable to the input key value. The key mask is in the same format as INKEY /K.

The format is:

```
@SELECT[filename,top,left,bottom,right,title[,sorted[,startline,[keymask]]]]
```

%@TIME - Added (not very useful, not recommended, and then only for the USA) support for am/pm time. For example:

```
%@TIME[1:39:15pm]
```

%@TALNUM[string] - Returns the number of alphanumeric (a-z, A-Z, and 0-9) characters in the string

%@TALPHA[string] - Returns the number of alphabetic characters (a-z, A-Z) in the string

%@TASCII[string] - Returns the number of 7-bit ASCII characters (0x00 - 0x7F) in the string

%@TCNTRL[string] - Returns the number of ASCII control characters (0x00 - 0x1F and 0x7F) in the string

%@TLOWER[string] - Returns the number of lower case alphabetic characters in the string

%@TUPPER[string] - Returns the number of upper case alphabetic characters in the string

%@TDIGIT[string] - Returns the number of decimal digits (0-9) in the string

%@TPRINT[string] - Returns the number of printable characters in the string

%@TPUNCT[string] - Returns the number of punctuation characters (printable characters which are not alphanumeric or space) in the string

%@TSPACE[string] - Returns the number of white space characters (0x09 - 0x0D or 0x20) in the string

%@TXDIGIT[string] - Returns the number of hexadecimal digits (0 - 9, A - F) in the string

Plugins:

Plugins can now access array variables directly through the ArrayVariables array. See TakeCmd.h in the SDK for details.

Updated Commands:**CD**

If the TCMD.INI directive "EverythingSearch" is set, CD will use "Everything Search" ([http://
www voidtools com](http://www voidtools com)) instead of JPSTREE.IDX for fuzzy directory searches. Everything Search is slightly faster, but will only work on local NTFS drives. Setting EverythingSearch is the equivalent of setting FuzzyCD=3 (***name***), unless you're using regular expressions.

CDD

If the TCMD.INI directive "EverythingSearch" is set, CDD will use "Everything Search" ([http://
www voidtools com](http://www voidtools com)) instead of JPSTREE.IDX for fuzzy directory searches. Everything Search is slightly faster, but will only work on local NTFS drives. Setting EverythingSearch is the equivalent of setting FuzzyCD=3 (***name***), unless you're using regular expressions.

COPY

If you specify the /C, /CF, /R, /U, or /UF options, COPY will append a ! to the copy specifier if the target exists and is being overwritten. For example:

```
[d:] copy file1 file2  
file1 =>! file2
```

If the EverythingSearch option is set, COPY won't try to update JPSTREE.IDX for local NTFS drives.

The /N option no longer creates empty subdirectories when used with /S.

DEL

If the EverythingSearch option is set, DEL won't try to update JPSTREE.IDX for local NTFS drives.

FREE

Now supports the OpenAFS 1.7.x redirector to retrieve disk space usage.

IF

If the "DupBugs" TCMD.INI directive (OPTION / Startup / "Duplicate CMD.EXE bugs") has been set, the IF behavior is different when in a command group in a batch file. If there are multiple command lines in the command group, a failed IF will now only ignore the remainder of the commands on that line. The commands on the subsequent lines will still be executed.

IFTP

/EP - Use Extended Passive mode. (Works with FTP and FTPS, but not SFTP.)

/IPv6 - By default, IFTP expects an IPv4 address for the local and remote host, and will create an IPv4 socket. The /IPv6 option tells IFTP to use IPv6 instead. (Works with FTP, FTPS, and SFTP connections.)

/PR="nnn" - When using active mode, IFTP uses any available port to listen to incoming connections from the server. You can override this behavior by setting /PR (PortRange) to a value containing the range of ports the class will be listening to. The range is provided as *start-end*, for instance: "1024-" stands for anything higher than 1024, "1024-2048" stands for ports between 1024 and 2048 inclusive, "4000-4010, 50000-50010" stands for ports between 4000 and 4010 or between 50000 and 50010. (Works with FTP and FTPS, but not SFTP.)

/Z[n] - Use Zlib compression. You can optionally set the compression level (0-9; the default is 7). Zlib compression must be enabled on the server, and will only work with FTP and FTPS connections (not SFTP).

JABBER

/F"filename" - Send a file to the specified target.

MD

If the EverythingSearch option is set, MD won't try to update JPSTREE.IDX.

MOVE

If you specify the /C, /CF, /R, /U, or /UF options, MOVE will append a ! to the move specifier if the target exists and is being overwritten. For example:

```
[d:\] move file1 file2  
file1 ->! file2
```

/G - Will now display the % moved even if Windows is doing a rename (which may be a copy & delete internally).

If the EverythingSearch option is set, MOVE won't try to update JPSTREE.IDX for local NTFS drives.

OPTION

OPTION now allows you to set "Auto SSL" for SMTP (i.e., SENDMAIL and SENDHTML).

OSD

OSD now lets you control up to 10 simultaneous OSD displays. (OSD allows you to create any number of windows, but you can only close the ones you've labeled from 0-9.) There are two new switches:

/C=n - Close the OSD window *n* (0-9). /C=n must be the only argument to OSD.

/ID=n - Open the OSD window *n* (0-9). /ID must be the first argument to OSD.

If you don't specify an /ID, OSD will default to window 0.

PLUGIN

PLUGIN now accepts multiple plugin name arguments. (The new syntax should still support

commands using the old syntax.) The syntax is:

```
PLUGIN [/B /C /F /I /K /L /P /U /V] plugin ...
```

PRINT

PRINT now accepts piped & redirected input to send to the printer. If there is no filename, PRINT will read from STDIN, create a temporary file, and send it to the printer.

PROMPT

~ - New metacharacter (substitute for P). If the environment variable HOME (or HOMEDRIVE + HOMEPATH) exists, TCC will compare the variable to the beginning of the current path. If they match, TCC will substitute ~ for the variable part. (If they don't match, ~ is treated like a P.)

For example:

```
[c:\] set home=c:\users\myself  
[c:\] set prompt=[$~]  
[c:\] cd \users\myself\downloads  
[~\downloads]
```

RD

If the EverythingSearch option is set, RD won't try to update JPSTREE.IDX for local NTFS drives.

REN

If the EverythingSearch option is set, REN won't try to update JPSTREE.IDX for local NTFS drives.

SENDHTML

/= (Command dialog) - Added the **BCC:** option.

/SMTP=server - Overrides the default SMTP server to use to send mail.

/USER=address - Overrides the default email account to use to send mail.

The OPTION command now allows you to set "Auto SSL" for SENDHTML.

SENDMAIL

/= (Command dialog) - Added the **BCC:** option.

/SMTP=server - Overrides the default SMTP server to use to send mail.

/USER=address - Overrides the default email account to use to send mail.

The OPTION command now allows you to set "Auto SSL" for SENDMAIL.

SYNC

If the EverythingSearch option is set, SYNC won't try to update JPSTREE.IDX for local NTFS drives.

TAR

/TEST - Test the integrity of the TAR file (header and contents). Any errors will be displayed on STDERR.

TPIPE

TPIPE is using a new version of the text pipe engine. There are a number of additional TPIPE options in v15.

Grep filters now allow Unicode patterns (when UTF-8 support mode is enabled).

Split filter now allows Unicode filenames, and Unicode file break patterns.

Removed the (completely useless) Quick Help from TPIPE. A "TPIPE /?" now invokes the online help for TPIPE.

/BUFFERSIZE - Sets the buffer size for the preceding search/replace filter. (The default is 4096.)

/buffersize=n

/EDITDISTANCE - Sets the edit distance threshold for the preceding search/replace filter. (The default is 2.)

/editdistance=n

/DATABASE - Adds a database-type filter.

/
database=Mode,GenerateHeader,Timeout,Connection,InsertTable,FieldDelimiter,Qualifier

Mode

- 0 Delimited output
- 1 Fixed width
- 2 XML
- 3 Insert script

GenerateHeader - Generates header information when True.

Timeout - SQL command timeout in seconds.

ConnectionString - The database connection string.

InsertTable - The name of the insert table.

FieldDelimiter - The string to use between columns.

Qualifier - The string to use around string column values.

/SELECTION - Added additional options for restriction filter types. (Restriction filters require sub filters to have any effect.)

```
/  
selection=Type,Locate,Param1,Param2,MoveTo,nDelimiter,CustomDelimiter,HasHeader[  
,ProcessIndividually]
```

The new *Type* options are:

- 1 Restrict lines
- 2 Restrict columns
- 3 Restrict to bytes
- 4 Restrict to delimited fields (CSV, Tab, Pipe etc.)

The new *ProcessIndividually* option specifies whether to apply sub filters to each CSV or Tab field individually (1), or to the fields as one string value (0). The default is false.

/MATHS - Adds a maths type filter. The syntax is:

```
/maths=operation,operand
```

operation - the operation to perform

- 0 +
- 1 -
- 2 *
- 3 div (the remainder is ignored)
- 4 mod (the remainder after division)
- 5 xor
- 6 and
- 7 or
- 8 not
- 9 shift left (0 inserted)
- 10 shift right (0 inserted)
- 11 rotate left
- 12 rotate right

operand - the operand to use

/PERL - Sets the Perl matching options for the immediately preceding search/replace filter.

```
/perl=BufferSize,Greedy,AllowComments,DotMatchesNewLines
```

BufferSize - The maximum buffer size to use for matches. Any match must fit into this buffer, so if you want to match larger pieces of text, increase the size of this buffer to suit. Default is 4096.

Greedy - If the pattern finds the longest match (greedy) or the shortest match. Default is false.

AllowComments - Allow comments in the Perl pattern. Default is false.

DotMatchesNewLines - Allow the '.' operator to match all characters, including new lines. Default is true.

/REPLACELIST - Add a search and replace list, using search and replace pairs from the specified file.

/

replacelist=Type,MatchCase,WholeWord,CaseReplace,PromptOnReplace,FirstOnly,SkipPromptIdentical,Simultaneous,LongestFirst,Filename

Type:

- 0 Replace
- 1 Pattern (old style)
- 2 Sounds like
- 3 Edit distance
- 4 Perl pattern
- 5 Brief pattern
- 6 Word pattern

MatchCase - Matches case when set to 1, ignores case when set to 0

WholeWord - Matches whole words only when set to 1

CaseReplace - Replaces with matching case when set to 1

PromptOnReplace - Prompts before replacing when set to 1

FirstOnly - If 1, only replace the first occurrence

SkipPromptIdentical - If 1, don't bother prompting if the replacement text is identical to the original.

Simultaneous - If 1, all search strings are scanned for simultaneously instead of consecutively. (This is useful if the search strings and results strings overlap.)

LongestFirst - If 1, searches for long phrases (most specific) before short phrases (least specific) - this is generally used for translations.

Filename - The file to load search/replace pairs from. If the file extension is .XLS or .XLSX, the file is assumed to be Excel format, if the extension is .TAB the file is assumed to have tab-delimited values, and any other extension (including .CSV) is assumed to have Comma-Separated Values.

The filename can contain environment variables enclosed in % signs e.g. %TEMP%\myfile.txt. TPIPE corrects any doubled backslashes.

/SCRIPT - Adds an ActiveX script filter. The syntax is:

/script=language,timeout,code

language: The language of the script

timeout: The command timeout in seconds

script: The code

/STARTSUBFILTERS - The following filters are created as sub filters, until the closing /

ENDSUBFILTERS. Sub filters allow a restricted part of the entire text to be operated on by a group of filters without effecting the entire text. For example, a "Restrict to delimited fields" (CSV, Tab, Pipe, etc.) filter can pick out a range of CSV fields, and then a search/replace filter can operate JUST on the text restricted.

/ENDSUBFILTERS - End the sub filters defined by the preceding /STARTSUBFILTERS.

UNTAR

/TEST - Test the integrity of the TAR file (header and contents). Any errors will be displayed on STDERR.

UNZIP

/TEST - Test the integrity of the ZIP file (header and contents). Any errors will be displayed on STDERR.

VIEW

VIEW now has the ability to view CSV files as tables. CSV files are typically used to represent tabular data, where each line in the file represents a row of a table. Each line contains the text of each column in the row, separated by a comma (although other characters can be used - e.g., a TAB).

By default, VIEW will automatically recognize CSV files and will display them as a table - where all the columns have the same width (much like a spreadsheet). Although unlike a spreadsheet, the column widths in V are fixed (determined by the longest entry in the column) and cannot be resized. You can press the arrow button next to the new CSV Mode button in the toolbar to customize the CSV behavior. Press the CSV Mode button to toggle between CSV mode and standard text mode.

ZIP

/TEST - Test the integrity of the ZIP file (header and contents). Any errors will be displayed on STDERR.

New Commands:

ASSOCIATE

Combines the ASSOC and FTYPE command. ASSOCIATE will display, delete, or create associations. The syntax is:

ASSOCIATE [/D /F /P /R filename /U] [.ext [program]]

/D - Delete the association for the specified .ext

/F - Force an overwrite of an existing association

/P - Pause after each page (only useful when running ASSOCIATE with no arguments)

/R - Read associations from a file. The lines in the file must be in the format
.ext=program

/U - Add the file association in HKCU instead of HKCR

DATEMONITOR

Monitor the current date and time, and execute the specified command when they match the saved time. If you don't specify any arguments, DATEMONITOR will display the current dates & times it is monitoring, and the associated commands.

DATEMONITOR [/C] yyyy-mm-dd hh:mm n command

/C	Clear any existing date monitors
n	Number of repetitions (or FOREVER)
command	Command to execute when the date matches the current time

The date must be in ISO (yyyy-mm-dd) format, and the time in 24-hour format.

DATEMONITOR sets two environment variables when the condition is triggered:

<u>_datemonitor</u>	The current date in yyyy-mm-dd format
<u>_timemonitor</u>	The current time in hh:mm (24-hour) format

ECHOX

Echo a line to STDOUT without performing any variable expansion or redirection. The syntax is:

ECHOX text

ECHOXERR

Echo a line to STDERR without performing any variable expansion or redirection. The syntax is:

ECHOXERR text

EVERYTHING

Search for files and/or directories on local NTFS drives using "Everything Search" (<http://www.voidtools.com>). EVERYTHING by default does a wildcard search equivalent to "**filename***", and outputs the full pathname of all matching files and/or directories. The syntax is:

EVERYTHING [/C /D /F /M=n /P /R /W] filename [...]

/C	Filename matching is case sensitive
-----------	-------------------------------------

/D	Only search for directories
-----------	-----------------------------

/F	Only search for files
-----------	-----------------------

/M=n	Only return a maximum of <i>n</i> files / directories. (Note that /M determines the total number of matches prior to any additional filtering. If you use /D or /F you will end up with the total minus the number of directories or files you excluded.)
-------------	---

- | | |
|----|--|
| /P | Match path names |
| /R | <i>filename</i> is a regular expression (EVERYTHING will automatically set the regular expression flag if the filename begins with ::) |
| /W | Match whole word |

You need to install Everything Search and index your local NTFS drives before using EVERYTHING.

SCREENMONITOR

Executes the specified command when a screen saver is active. If you don't specify any arguments, SCREENMONITOR will display the current screen saver monitor command (if any). Once the condition has been set, it will not be set again until the screen saver becomes inactive and then active again.

SCREENMONITOR [/C] n command

- | | |
|----------------|---|
| /C | Clear any existing screen saver monitors |
| n | Number of repetitions (or FOREVER) |
| command | Command to execute when the screen saver becomes active |

2.2 Version 14

Feature List:

Take Command now supports a splitter window (on the horizontal scrollbar). You must enable "Splitter Windows" in the Take Command configuration dialog (Tabs window), and restart TCMD to see the splitter. (Note that it is technically impossible to display splitter console windows, so TCMD is using a lot of hand-waving, smoke, and mirrors.) The splitter window (on the right side) will not automatically scroll to the end when new output is displayed, or when you enter new commands. This allows you to scroll back in the screen buffer to review previous commands and output, and to select text from previous pages.

Take Command will check to see if your maximum allowable console window size (as set by Windows) is smaller than your Take Command tab window; if so Take Command will reduce the console font size until the console window size matches the tab window. (Requires Windows Vista or later.)

The Internet code has been substantially rewritten and ported to a new major update of the IPWorks dll's.

Updated the IDE editor to a new version of Scintilla (3.2).

The conditional tests (DO, IF, IFF, etc.) now accept ! as a synonym for NOT.

Alt-F9 will restore the original filename mask when doing filename completion. This will only work provided you haven't terminated the completion loop; i.e., by pressing anything other than tab, F8, F9, F10, or F12.

Alt-F6 will no longer open the Folder View and List View windows if they're disabled; it will toggle

between the Command Input window and the active tab window.

Added a "Register for all users" option (checkbox on the register page). This option will only be enabled if you are running an elevated administrator session.

There will not be a TCC/LE 14.0.

Batch Editor / Debugger:

You can select a rectangular area by holding down the Alt key while clicking the left mouse button and dragging the mouse.

INI Directives:

CompleteAllFiles=yes|NO - Normally, **TCC** will only complete directories and executable files (as defined by PATHEXT) when you press Tab or F9 at the beginning of a command line. If CompleteAllFiles is set to YES, **TCC** will complete any matching filename. Note that if you also have CompletePaths set, you'll probably have several hundred (or thousand!) matches for any filename you enter.

SplitterWindows=NO|yes - If YES, Take Command will display a horizontal scrollbar with a splitter in each tab window.

Internal Variables:

[SERIALPORTS](#) - Returns a space-delimited list of all of the available serial ports (COM1 - COMN).

Variable Functions:

[@FILES\[/H filename\]](#) - Don't count "." or ".."

[@REREPLACE\[source_re, target_re, source \]](#) - Regular expression back reference replacement.

source_re - Regular expression to apply to the source

target_re - Regular expression for back reference

source - Source string

[@SERIALPORTCLOSE\[n \]](#) - Close the serial port. "n" is the handle returned by [@SERIALPORTOPEN](#).

[@SERIALPORTFLUSH\[n \]](#) - Flush the contents of the serial port buffer. "n" is the handle returned by [@SERIALPORTOPEN](#).

[@SERIALPORTOPEN\[COMn\[, baud\[, parity\[, bits\[, flow\]\]\]\]\]](#) - Open a serial port for read & write. The parameters are:

COMn - The COM port to open (COM1 - COM9)

baud - The baud rate (110 - 256000)

parity - The parity scheme to use. This can be one of the following values:

no
odd
even
mark
space

bits - The number of bits in the bytes to transmit & receive

flow - The type of flow control to use. This can be one of the following values:

no
CtsRts
CtsDtr
DsrRts
DsrDtr
XonXoff

@SERIALPORTOPEN returns a handle to the serial port, which must be passed to the other serial port functions.

[@SERIALPORTREAD](#)[n] - Reads a string from the serial port. "n" is the handle returned by @SERIALPORTOPEN.

[@SERIALPORTWRITE](#)[n, text] - Writes a string to the serial port. "n" is the handle returned by @SERIALPORTOPEN.

[@SMCLOSE](#)[n] - Close a shared memory handle.

n - The shared memory handle returned by @SMOPEN

[@SMOPEN](#)[size, name] - Open a handle to shared memory

size - The size of shared memory (in bytes)

name - The name of the shared memory. The name can have a "Global\" or "Local\" prefix to create the object in the global or session namespace.

[@SMPEEK](#)[*handle,offset,size*] - Read a value from shared memory.

handle - a handle from @SMOPEN

offset - the byte offset in the buffer (decimal or hex)

size - the size of the value to read (in bytes):

1 - character
2 - short
4 - int
8 - int64

[@SMPOKE](#)[*handle,offset,size,value*] : Write a value to shared memory

handle - a handle from @SMOPEN

offset - the byte offset in the buffer (decimal or hex)

size - the size of the value (in bytes):

- 1 - character
- 2 - short
- 4 - int
- 8 - int64

value - the value to poke

@SMREAD[n, offset, type, length] - Read a string from shared memory

n - The shared memory handle returned by @SMOPEN

offset - The offset (in bytes) from the beginning of the shared memory buffer.

type - Either **a** to read the string as ASCII or **u** to write it as Unicode.

length - The length of the string (in characters) to read.

@SMWRITE[n, offset, type, string] - Write a string to shared memory

n - The shared memory handle returned by @SMOPEN

offset - The offset (in bytes) from the beginning of the shared memory buffer.

type - Either **a** to write the string as ASCII or **u** to write it as Unicode.

string - The string to write.

@TRIMALL[string] - Remove leading and trailing spaces and tabs, and extra internal spaces and tabs.

Updated Commands:

ATTRIB

/L - Set or display the attributes of the symbolic link versus the target of the symbolic link.

COPY

If you don't specify any arguments, COPY will display the command dialog.

Added support for regular expression back references in the target name. If you are using back references, you must use a regular expression in the source name. The syntax is:

```
copy ::filename ::target
```

See the help for details about back references.

DEL

If you don't specify any arguments, DEL will display the command dialog.

DESCRIBE

If you don't specify any arguments, DESCRIBE will display the command dialog.

ESET

/C - copy the value from another variable / alias / function. The syntax is:

```
eset /c var1 var2
```

where "var1" is the variable whose value you want to copy, and "var2" is the variable (new or existing) that you want to update.

GLOBAL

If you don't specify any arguments, GLOBAL will display the command dialog.

JABBER

If you don't specify any arguments, JABBER will display the command dialog.

MD

If you don't specify any arguments, MD will display the command dialog.

MKLINK

If you don't specify any arguments, MKLINK will display the command dialog.

MKLNK

If you don't specify any arguments, MKLNK will display the command dialog.

MOVE

If you don't specify any arguments, MOVE will display the command dialog.

Added support for regular expression back references in the target name. If you are using back references, you must use a regular expression in the source name. The syntax is:

```
move ::source ::target
```

See the help for details about back references.

PDIR

Now supports multiple nested *'s in a @ function specification.

PLAYSOUND

If you don't specify any arguments, PLAY SOUND will display the command dialog.

RD

If you don't specify any arguments, RD will display the command dialog.

REN

If you don't specify any arguments, REN will display the command dialog.

Added support for regular expression back references in the target name. If you are using back references, you must use a regular expression in the source name. The syntax is:

```
ren ::source ::target
```

See the help for details about back references.

SELECT

If you don't specify any arguments, SELECT will display the command dialog.

SENDHTML

If you don't specify any arguments, SENDHTML will display the command dialog.

SENDMAIL

If you don't specify any arguments, SENDMAIL will display the command dialog.

SET

/RO var=value - set a read-only variable. Once you've set the variable, you cannot change it (or unset it). Only environment variables can be read-only (not registry variables or array variables).

SETARRAY

/R filename arrayname - read a file into a (1-dimensional) array. (SETARRAY will determine the required size of the array.)

START

/Desktop=desktopname - specify the desktop where you want to start the application.

/NODE n - Start the program using the specified NUMA node (*n* is a decimal integer).

/TABNA - start a new Take Command tab window, but keep the current tab active.

SYNC

If you don't specify any arguments, SYNC will display the command dialog.

TAR

If you don't specify any arguments, TAR will display the command dialog.

TIMER

Accepts an optional command to run. This is the equivalent of "timer on & command & timer off". The syntax is:

timer command [args]

TOUCH

If you don't specify any arguments, TOUCH will display the command dialog.

UNTAR

If you don't specify any arguments, UNTAR will display the command dialog.

UNZIP

If you don't specify any arguments, UNZIP will display the command dialog.

ZIP

If you don't specify any arguments, ZIP will display the command dialog.

New Commands:DEBUGMONITOR

Monitors the OutputDebugString API call from any process. The syntax is:

DEBUGMONITOR [/C]
DEBUGMONITOR n command

DEBUGMONITOR will set the environment variable "_outputdebugstring" to the string specified in the OutputDebugString call.

DESKTOP

Create a new desktop or switch to an existing desktop. The syntax is:

DESKTOP /C [/N] newdesktopname - create and optionally switch to a new desktop
DESKTOP desktopname - switch to an existing desktop

If you don't specify any arguments, DESKTOP will display the existing desktops.

RESOLUTION

Change the resolution (and optionally the color depth and refresh frequency) of the specified display. The syntax is:

RESOLUTION [displayname] width height [depth [frequency]]

If you don't specify any arguments, RESOLUTION will display the display devices and monitors.

TPIPE

Text filtering and substitution. You can specify multiple filters, which are processed in the order they appear on the command line. Do not insert any unquoted whitespace in the arguments to an option! Row and column positions start at 1.

The syntax is:

```
TPIPE [/input=filename] [/output=filename] [/filter=filename] [/unicode=input,output] [/
save=filename] [/simple=n[u]] [/eol=input,output,length] [/
line=start,increment,skipblank,dontnumberblank,format] [/insert=position,type,string] [/
head=Exclude,LinesOrBytes,Count] [/tail=Exclude,LinesOrBytes,Count] [/number=type,value] [/
string=type,string] [/file=type,filename] [/
dup=RemoveDuplicateLines,IgnoreCase,StartColumn,Length,IncludeOne] [/comment=text] [/
log=LogFileName] [/run=InputFileName,OutputFileName,"CommandLine"] [/
merge=type,filename] [/
split=type,SplitSize,SplitChar,SplitCharPos,SplitCharCount,SplitLines,SplitFilename] [/
grep=Type,IncludeLineNumbers,IncludeFilename,IgnoreCase,CountMatches,UTF8,PatternType,
Pattern] [/
replace=Type,MatchCase,WholeWord,CaseReplace,PromptOnReplace,Extract,FirstOnly,SkipPr
omptIdentical,Action,SearchStr,ReplaceStr] [/
xml=Type,IncludeText,IncludeQuotes,MatchCase,BufferSize,Tag,Attribute,EndTag]
```

/input=filename

Filename to read. This can be either a disk file, include file (@filename), or CLIP:. If it is not specified, TPIPE will read from standard input.

/output=filename

Filename to write. This can be either a disk file or CLIP:. If it is not specified, TPIPE will write to standard output.

/merge=type,filename

Adds a merge type filter (merge into single output filename). The arguments are:

type:

- 0 Merge into filename
- 1 Retain lines found in filename
- 2 Remove lines found in filename
- 3 Link filter filename

filename - the filename to use

/filter=filename

Name of filter file to load (see /save=filename)

/save=filename

Saves the filter settings defined on the command line to the specified filename, and returns without executing any filters.

/unicode=input,output

Convert the file to or from Unicode. **input** is the encoding for the input file; **output** is the encoding for the output file. The possible values are:

UTF-16LE
UTF-16BE
UTF-32LE
UTF-32BE
UTF-8
ANSI
ASCII
CP nnn , where nnn is the Windows code page.

TPIPE handles files internally as UTF-8, so if you want to process a Windows UTF-16LE file, you'll need to convert it to UTF-8 first, then apply the desired filters, and convert it back to UTF-16LE.

/simple=n[u]

Adds a simple filter type. n is the type of filter to add, and for those filters that support it, u indicates that the filter will be dealing with Unicode data.

- 1 Convert ASCII to EBCDIC
- 2 Convert EBCDIC to ASCII
- 3 Convert ANSI to OEM
- 4 Convert OEM to ANSI
- 5 Convert to UPPERCASE
- 6 Convert to lowercase
- 7 Convert to Title Case
- 8 Convert to Sentence Case
- 9 Convert to tOGGLE cASE
- 10 Remove blank lines
- 11 Remove blanks from End of Line
- 12 Remove blanks from Start of Line
- 13 Remove binary characters
- 14 Remove ANSI codes
- 15 Convert IBM drawing characters
- 16 Remove HTML and SGML
- 17 Remove backspaces
- 18 Resolve backspaces
- 19 Remove multiple whitespace
- 20 UUEncode
- 21 Hex Encode
- 22 Hex Decode
- 23 MIME Encode (Base 64)
- 24 MIME Decode (Base 64)

- 25 MIME Encode (Quoted printable)
- 26 MIME Decode (Quoted printable)
- 27 UUDecode
- 28 Extract email addresses
- 29 Unscramble (ROT13)
- 30 Hex dump
- 32 XXEncode
- 33 XXDecode
- 34 Reverse line order
- 35 Remove email headers
- 36 Decimal dump
- 37 HTTP Encode
- 38 HTTP Decode
- 39 Randomize lines
- 40 Create word list
- 41 Reverse each line
- 42 Convert to RanDom case
- 43 Extract URLs
- 44 ANSI to Unicode
- 45 Unicode to ANSI
- 46 Display debug window
- 47 Word concordance
- 48 Delete all
- 49 Restrict to each line in turn
- 50 Convert CSV to Tab-delimited
- 51 Convert CSV to XML')
- 52 Convert Tab-delimited to CSV
- 53 Convert Tab-delimited to XML
- 54 Convert CSV (with column headers) to XML
- 55 Convert Tab-delimited (with column headers) to XML
- 56 Convert CSV (with column headers) to Tab-delimited
- 57 Convert Tab-delimited (with column headers) to CSV
- 58 Restrict to file name
- 59 Convert Word documents to text
- 60 Swap UTF-16 word order
- 61 Swap UTF-32 word order
- 62 Remove BOM (Byte Order Mark)
- 63 Make Big Endian
- 64 Make Little Endian
- 65 Compress to Packed Decimal
- 66 Compress to Zoned Decimal
- 67 Expand Binary Number to EBCDIC
- 68 Expand Binary Number to ASCII
- 69 NFC - Canonical Decomposition, followed by Canonical Composition
- 70 NFD - Canonical Decomposition
- 71 NFKD - Compatibility Decomposition
- 72 NFKC - Compatibility Decomposition, followed by Canonical Composition
- 73 Decompose
- 74 Compose
- 75 Convert numeric HTML Entities to text
- 76 Convert PDF documents to text
- 77 Restrict to ANSI files
- 78 Restrict to Unicode UTF16 files

79 Restrict to Unicode UTF32 files
80 Convert Excel spreadsheets to text

/eol=input,output,length

Add an EOL (end of line) conversion filter. The arguments are:

input:

- 0 - Unix (LF)
- 1 - Mac (CR)
- 2 - Windows (CR/LF)
- 3 - Auto
- 4 - Fixed (use the length parameter to specify the length)

output:

- 0 - Unix
- 1 - Mac
- 2 - Windows
- 3 - None

length - The line length to use if input=4

/line=StartNumber,Increment,SkipBlank,DontNumberBlank,NumberFormat

Adds a Line Number filter. The arguments are:

StartNumber - the starting line number

Increment - the amount to add for each new line number

SkipBlankIncrement - don't increase the line number for blank lines

DontNumberBlank - don't put a line number on blank lines

NumberFormat - The format to use for the line number. The format syntax is:

[-][width][.precision]d

An optional left justification indicator, ["-"]

An optional width specifier, [width] (an integer). If the width of the number is less than the width specifier, it will be padded with spaces.

An optional precision specifier [precision] (an integer). If the width of the number is less than the precision, it will be left padded with 0's.

The conversion type character:

d - decimal

/insert=position,type,string

Add an insert type filter. The arguments are:

type:

- 0 - Insert column
- 1 - Insert bytes

position - the position to insert the string

string - the string to insert

/head=Exclude,LinesOrBytes,Count

Add a head type filter (includes or excludes text at the beginning of the file). The arguments are:

Exclude - if 0, include the text; if 1, exclude it

LinesOrBytes - if 0, measure in lines; if 1, measure in bytes

Count - the number of lines or bytes to include or exclude

/tail=Exclude,LinesOrBytes,Count

Add a tail type filter (includes or excludes text at the end of the file). The arguments are:

Exclude - if 0, include the text; if 1, exclude it

LinesOrBytes - if 0, measure in lines; if 1, measure in bytes

Count - the number of lines or bytes to include or exclude

/dup=Type,IgnoreCase,StartColumn,Length,IncludeOne

Remove or show duplicate lines. The arguments are:

Type:

- 0 - Remove duplicate lines
- 1 - Show duplicate lines

IgnoreCase - if 1, ignore case during comparisons

StartColumn - The starting column for comparisons

Length - The Length of the comparison

IncludeOne - Include lines with a count of 1

/string=type,MatchCase,string

Add a string-type filter. The arguments are:

type:

- 0 Add left margin
- 1 Add header
- 2 Add footer
- 3 Add right margin
- 4 Remove lines
- 5 Retain lines
- 6 Remove lines matching perl pattern
- 7 Retain lines matching perl pattern

- 8 Add text side by side
- 9 Add repeating text side by side
- 10 Not Used
- 11 Not Used
- 12 XSLT transform
- 13 Restrict to lines from list
- 14 Restrict to lines NOT in list
- 15 Restrict to lines matching perl pattern
- 16 Restrict to lines NOT matching perl pattern

matchCase - case sensitive or not (where appropriate)

string - the string to use

/file=type,MatchCase,filename

Add a file-type filter. The arguments are:

type:

- 17 Restrict to filenames matching perl pattern
- 18 Restrict to filenames NOT matching perl pattern

MatchCase - If 1, do a case sensitive match (where appropriate)

filename - the filename to use

/number=type,value

Add a number-type filter. The arguments are:

type:

- 0 Convert Tabs to Spaces
- 1 Convert Spaces to Tabs
- 2 Word wrap (value column width)
- 3 Pad to width of value
- 4 Center in width of value
- 5 Right justify in width of value
- 6 Restrict CSV field to value
- 7 Restrict tab-delimited field to value
- 8 Truncate to width value
- 9 Force to width value
- 10 Repeat file value times
- 11 Restrict to blocks of length
- 12 Expand packed decimal (with implied decimals)
- 13 Expand zoned decimal (with implied decimals)
- 14 Expand unsigned (even-length) packed decimal
- 15 Expand unsigned (odd-length) packed decimal

Value - the numeric value to use

/comment=text

Add a comment to a filter file.

Text - Comment to add

/log=Filename

Log the TPIPE actions.

Filename - Name of log file

/run=InputFileName,OutputFileName,"CommandLine"

Adds a Run External Program filter. The arguments are:

InputFilename - the filename that TextPipe should read from after the External Program writes to it.

OutputFilename - the filename that TextPipe should write to for the External Program to read in.

CommandLine - the command line of the program to run. Should include double quotes around the entire command line.

/split=type,SplitSize,SplitChar,SplitCharPos,SplitCharCount,SplitLines,SplitFilename

Adds a split type filter. The arguments are:

type:

- 0 Split at a given size
- 1 Split at a given character
- 2 Split at a given number of lines

splitSize - the size file to split at

splitChar - the character to split at

splitCharPos -

- 0 Split before the character (it goes into the next file)
- 1 Split after the character (it remains in the first file)
- 2 Split on top of the character (remove it)

SplitCharCount - the number of times to see SplitChar before splitting

SplitLines - (optional) split after a given number of lines, default 60

SplitFilename - (optional) the name to give to each output split file. /split will append a "%3.3d" format specifier to the name; i.e. an input file "foo.txt" will generate output files named "foo.txt.000", "foo.txt.001", etc. If you don't specify a SplitFilename, /split will use the input filename as the base.

/grep=Type,IncludeLineNumbers,IncludeFilename,IgnoreCase,CountMatches,PatternType,UTF8,IgnoreEmpty,Pattern

Adds a Grep type line based filter. The arguments are:

Type:

- 0 Restrict lines matching
- 1 Restrict lines NOT matching
- 2 Extract matches
- 3 Extract matching lines (grep)
- 4 Extract non-matching lines (inverse grep)
- 5 Remove matching lines
- 6 Remove non-matching lines

IncludeLineNumbers - 1 to include the line number where the pattern was found

IncludeFilename - 1 to include the filename where the pattern was found

IgnoreCase - 1 to ignore case when matching the pattern

CountMatches - 1 to only output a count of the number of matches

PatternType

- 0 Perl pattern
- 1 Egrep pattern
- 2 Brief pattern
- 3 MS Word pattern

UTF8 - 1 to allow matching Unicode UTF8 characters

IgnoreEmpty - 1 to ignore empty matches

Pattern - the (regular expression) pattern to match

/

**replace=Type,MatchCase,WholeWord,CaseReplace,PromptOnReplace,Extract,First
Only,SkipPromptIdentical,Action,SearchStr,ReplaceStr**

Adds a search and replace (find and replace) filter. The arguments are:

Type:

- 0 Replace
- 1 Pattern (old style)
- 2 Sounds like
- 3 Edit distance
- 4 Perl pattern
- 5 Brief pattern
- 6 Word pattern

MatchCase - Matches case when set to 1, ignores case when set to 0

WholeWord - Matches whole words only when set to 1

CaseReplace - Replaces with matching case when set to 1

PromptOnReplace - Prompts before replacing when set to 1

Extract - If 1, all non-matching text is discarded

FirstOnly - If 1, only replace the first occurrence

SkipPromptIdentical - If 1, don't bother prompting if the replacement text is identical to the original.

Action - the action to perform when found:

- 0 replace
- 1 remove
- 2 send to subfilter
- 3 send non-matching to subfilter
- 4 send subpattern 1 to subfilter etc

SearchStr - the string to search for

ReplaceStr - the string to replace it with

/xml=Type,IncludeText,IncludeQuotes,MatchCase,BufferSize,Tag,Attribute,EndTag

Adds an HTML/XML filter. The arguments are:

Type - the operation to perform:

- 0 restrict to an element
- 1 restrict to an attribute
- 2 restrict to between tags

IncludeText - whether to include the find string in the restriction result (default false)

IncludeQuotes - whether to include surrounding quotes in the attribute result or not (default false)

MatchCase - match case exactly or not (default false)

BufferSize - the maximum expected size of the match (default 32768)

Tag - the element or start tag to find

Attribute - the attribute to find

EndTag - the endTag to find

2.3 Version 13

NEW VERSION OVERVIEW - Take Command 13.04

VIEW - The menus and dialogs are now available in English, French, Russian, and Spanish. The first time you run VIEW, it will set the language to the one defined in TCMD.INI. If you want to change it, you can select a language by starting VIEW, and selecting Tools / Preferences from the menu.

Updated all of the Internet libraries (ip*.dll).

NEW VERSION OVERVIEW - Take Command 13.03

Updated all of the Internet libraries (ip*.dll).

Updated zip library.

Updated registration (licensing) library.

New installer version.

Added Google Translate to web help.

MKLINK - /D will now create a directory symlink even if the directory to link to does not exist (for CMD compatibility).

NEW VERSION OVERVIEW - Take Command 13.02

The batch debugger editor and command input editor have been upgraded to a major new version.

Added the feedback tab to the web help pages.

BDEBUGGER - Added the Ctrl-F3, F3, and Shift-F3 keys (find next / find prev).

NEW VERSION OVERVIEW - Take Command 13.01

The local help (tcmd.chm) has a number of new display options.

Take Command and TCC now have the option to use either the new Web Help or the local help (tcmd.chm). The web help has some additional features, including the ability to add comments to help topics.

The search algorithm for the popup windows has changed slightly. TCC will append a * to the search string unless the last character of the string is a wildcard (?*, *, or []), or if the search string is a regular expression (the first two characters of the string are ::).

MKLINK - The /X option no longer requires two arguments.

[Take Command/LE has been discontinued](#). Existing Take Command/LE customers have been sent a free update to the full Take Command version.

.INI Directives:

WebHelp=yes|NO - New TCMD.INI directive (set in the Take Command and the TCC configuration dialogs). If set,

Take Command and TCC will use the web help at <http://jpsoft.com/help/index.htm> instead of the local help.

NEW VERSION OVERVIEW - Take Command 13.0

This is a summary of the new features. For complete details, see the appropriate topics in this help file. Features marked with a * are also in TCC/LE; all other features are Take Command and/or TCC

only.

Feature List:

- * TCC/LE is now available in 32-bit and 64-bit versions.
- * Removed the (non-Windows) limits on filename and argument size. The maximum filename size (dictated by Windows) and single argument size is now 32,767 characters.
- * The maximum (input) command line is now 65,535 characters.
- * The maximum expanded command line size is now 131,072 characters.
- * Fuzzy directory (jpstree.idx) updates (in MD, RD, DEL /S/X, etc.) are twice as fast as v12.

The Take Command toolbar has a new button "V" that will call the new file viewer VIEW (see below).

All of the popup windows (history lists, filename completion, @select, etc) have an edit control on the toolbar. Entering a search string there (or just typing while the popup window has focus) will eliminate non-matching entries from the window. The search string can also contain wildcards or regular expressions. (For example, entering *jpsoft* in the edit control at the top of the window will select all matching lines that contain "jpsoft" anywhere.)

All paging options (/P) in TCC now accept an "All" key to turn off paging. (This will vary by language, and is shown in the prompt.)

TCC now sets the default regular expression syntax (according to the value set in TCMD.INI) at startup and after saving new values in OPTION. (This is only of interest to plugin authors.)

The persistent directory history file is now only loaded if either local directory history is set, or TCC thinks it is the first instance. But since there's no such thing as "shell levels" in Windows, TCC can only guess (based on the TCMD.INI inheritance).

If the DirHistoryOnEntry .INI directive is set, TCC will save its startup directory to the directory history.

A Ctrl-Enter in the Command Input window will execute the current line and move the cursor down to the next line (without inserting a line).

Take Command now saves toolbar changes done through the add tab or create/edit button dialogs (previously it required you to right click in the toolbar & select "Save to .INI").

Help file updates.

.INI Directives:

AliasSize - Set the size of the global alias list (in characters). (The default is 256K.)

BeepFreq / BeepLength-- you can play a system sound on an error by setting BeepLength to 0 and BeepFreq to the desired sound:

- 0 - Windows default beep sound
- 16 - Windows Critical Stop sound

32 - Windows Question sound
48 - Windows Exclamation sound
64 - Windows Asterisk sound

DelWipePasses - The number of passes for a DEL /W. The range is 1-9; the default is 3.

FunctionSize - Set the size of the global function list (in characters). (The default is 128K.)

Command Line Editing:

A Ctrl-Enter in the Command Input window will execute the current line and move the cursor down to the next line (without inserting a line).

An Alt-F1 will look to see if the (first and only) argument on the line is an internal command with a command dialog (see below), and if so invoke the dialog.

The popup history windows have an edit control on the toolbar. Entering a search string there (or just typing while the popup window has focus) will eliminate non-matching entries from the window.

New Commands:

[SENDHTML](#)

Send an HTML email.

SENDHTML [/A file1 [/A file2 ...] /D /Eaddress /H"header: value" /In /M /Pn /R /Sn /SSL[=n] /V "address[,address...]" [cc:address[,address] bcc:address[,address...]]" subject [text | @msgfile]

file1...	The attachment files
address	The destination email address
subject	The subject line
text	The message to send
msgfile	The file containing the message body
/SSL=n	SSL negotiation type

/A file	Attachment
/D	Delivery Confirmation
/E	Reply-to address
/H	Send custom header
/In	Importance
/M	CRAM-MD5 authentication
/Pn	Priority
/R	Send read receipt
/Sn	Sensitivity
/V	Verbose
/X	Send EHLO instead of HELO

If you pass "/=" as the argument, SENDHTML will display a dialog to help you with the command line options.

[TCDIALOG](#)

Run the new internal [command dialogs](#). See the Commands section for details.

The syntax is:

TCDIALOG command

If "command" does not have a dialog, TCDIALOG returns a usage error.

[VIEW](#)

VIEW is a replacement for LIST. The syntax is:

VIEW [/A:[[-+]rhsadecijopt /A /B /E /F /FIX/FLAT / GB /H /L /L:n ?LEN:n /O:xx /P /R /S:xx /T /TEXT /VH /W] file ...]

- /= Display the VIEW command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.
- /A: Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with [@file lists](#). See [@file lists](#) for details.
 - You can specify /A:= to display a dialog to help you set individual attributes.
- /A View the file in ASCII mode. This is the default mode and will only need to be specified in order to override an existing EBCDIC mode.
- /B View the file in EBCDIC mode. VIEW normally automatically determines if a file is EBCDIC and automatically sets this mode.
- /E Start viewing the file from the end instead of the beginning.
- /F Standard input has been redirected (such as the output of a DIR command). (VIEW can normally determine this on its own.)
- /FIX:n When viewing a file, the display may be fixed at a certain column position so that any text to the left of the fixed column will always be visible (ie, it will not scroll off the screen).
- /FLAT Enables Flat Text Mode. This is a cross between text and hex modes. The file is displayed as text, however, control characters like line feeds and tabs are not expanded, and the file is always wrapped at the specified wrap length.
- /GB Enables Greenbar Mode. (Each alternating line is in a different color.)
- /H View the file in Hex mode.
- /L Display the last file that was viewed. (This will be the first file in the Recent Files list.)
- /L:n Start displaying the file from line number *n*. A solid blue line will appear at the top of the file, indicating that a non-zero start offset is being used.

- /LEN:n Set the wrap length to *n*.
/O:xx Start displaying the file from offset *xx*.
/P Print the file and exit VIEW when finished.
/R When started with no parameters, VIEW will browse the current directory. By specifying the /R option, VIEW will display the directory that it last browsed.
/S The /S option is used to tell VIEW to start displaying the file at the position of a string match. The format of the /S command line option is as follows:

/S:*SearchString* /SO:[CWRHUB] /SN:*n* /SC:Columns

where SO can contain a series of letters which correspond to the options in the search dialog box. These can be one of:

- C** Match case
W Word Only
R Regular Expression
H Hex/Binary
U Unicode
B Search backwards (from end of file)

SN indicates which occurrence of the string to find. By default, the first match is found (*n*=1).

SC can be used to restrict the search to certain columns.

If the search string contains spaces, you must enclose it in double quotes.

- /T Enable File Tailing. If data is added to the file while you are viewing it, it will automatically be updated. There is no need to press the Refresh button to see any changes since the file was loaded. This is particularly useful when viewing log files while they are still being updated.
/TEXT Open the files in text mode (opposite of /H). (This is the default.)
/VH Display the file in Vertical Hex Mode. This is a cross between Text and Hex modes. The file is displayed one line at a time (just as in text mode). However, each line is followed by 2 lines containing the hex code of each character in the line.
/W Display the VIEW window in a Take Command tab window.

WEBFORM

POST data to interactive web pages or scripts. WEBFORM will use the proxy & firewall settings

from TCMD.INI.

WEBFORM [/An /En /Fn /U"username" /P"password" /R"referer" /V] /W"url" "varname"
"varvalue" ...

/An Authorization scheme:
0 - basic
1 - digest
2 - proprietary
3 - none
4 - NTLM
5 - Negotiate

/En Encoding:
0 (URLEncoding) This is the most common encoding for HTML form contents.
1 (MultipartFormData) This is MIME encoding allowing transmission of binary data.
2 (QueryString) This is an older form of encoding where the actual parameters are appended to the URL query string. (Generally not recommended because most servers limit the size of the URL to less than 1K or 2K).

/F"from" Email address of the HTTP agent.

/U"username" User name if authentication is to be used.

/P"password"

/L"localfile" Local file for downloading. If the file exists, it will be overwritten.

/O"headers" Other headers. The headers must be of the format "header: value" as described in the HTTP specifications. Header lines should be separated by CR/LF (^r^n).

/R"referer" The document referring the requested URL

/Tn Firewall type:
0 - no firewall (default)
1 - Connect through a tunneling proxy. Port is set to 80.
2 - Connect through a SOCKS4 proxy. Port is set to 1080.
3 - Connect through a SOCKS5 proxy. Port is set to 1080.

/V(bose) Display retrieved document text

/W"url" URL of web page

Example:

```
webform /v /w"http://download.finance.yahoo.com/d/quotes.csv" "f", "sl1d1t1c1ohgv"
"e", ".csv" "s", "IBM"
```

WEBUPLOAD

Upload files to RFC1867-compliant web servers. WEBUPLOAD will use the proxy & firewall settings from TCMD.INI.

WEBUPLOAD [/An /Fn /U"username" /P"password" /R"referer" /V] /W"url" [/V "varname"

"varvalue"] "filevar" "filename" ...

/An Authorization scheme:

- 0 - basic
- 1 - digest
- 2 - proprietary
- 3 - none
- 4 - NTLM
- 5 - Negotiate

/F"from" Email address of the HTTP agent.

/U"username" User name if authentication is to be used.

/P"password"

/L"localfile" Local file for downloading. If the file exists, it will be overwritten.

/O"headers" Other headers. The headers must be of the format "header: value" as described in the HTTP specifications. Header lines should be separated by CR/LF (^r^n).

/R"referer" The document referring the requested URL

/Tn Firewall type:

- 0 - no firewall (default)
- 1 - Connect through a tunneling proxy. Port is set to 80.
- 2 - Connect through a SOCKS4 proxy. Port is set to 1080.
- 3 - Connect through a SOCKS5 proxy. Port is set to 1080.

/V The following two arguments are a varname / varvalue pair.

Commands:

ACTIVATE

DESKTOP - makes the desktop the active window

ASSOC

/U - display/set the associations in HKCU\Software\Classes

ATTRIB

If you pass "/=" as the argument, ATTRIB will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, ATTRIB will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, ATTRIB will display a dialog to help you set the individual attributes.

Sets three internal variables:

%_attrib_dirs - The number of directories modified
%_attrib_files - The number of files modified
%_attrib_errors - The number of errors

CD

Fuzzy directory searches are 250% faster.

Extended the ~ (home) path argument to take appended directory names (i.e., "~\music"). If CD cannot find HOME in the environment, it will look for HOMEDRIVE + HOMEPATH.

CDD

Fuzzy directory searches are 250% faster.

Extended the ~ (home) path argument to take appended directory names (i.e., "~\music"). If CDD cannot find HOME in the environment, it will look for HOMEDRIVE + HOMEPATH.

COPY

If you pass "/=" as the argument, COPY will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, COPY will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, COPY will display a dialog to help you set the individual attributes.

Sets three internal variables:

%_copy_dirs - The number of directories created
%_copy_files - The number of files copied
%_copy_errors - The number of errors

DEL

If you pass "/=" as the argument, DEL will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, DEL will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, DEL will display a dialog to help you set the individual attributes.

/W[n] - takes a new option to specify the number of wipe passes. The range is 1-9999; the default is 3. (It will be REALLY slow and probably really pointless at anything over 3-5.) See also the .INI directive DelWipePasses.

Sets three internal variables:

%_del_dirs - The number of directories deleted
%_del_files - The number of files deleted
%_del_errors - The number of errors

DELAY

UNTIL [yyyy-mm-dd] hh:mm[:ss] - delay until the specified date/time. If no date is specified, default to today.

DESCRIBE

If you specify "/[=]" for the ranges, DESCRIBE will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, DESCRIBE will display a dialog to help you set the individual attributes.

DIR

If you pass "/=" as the argument, DIR will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, DIR will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, DIR will display a dialog to help you set the individual attributes.

Sets three internal variables:

- %_dir_dirs - The number of directories displayed
- %_dir_files - The number of files displayed
- %_dir_errors - The number of errors

/B1 - display bare filenames with the relative path from the start, when used with /S. (Normally, /B shows the full pathname for each file.)

DIRHISTORY

If you pass "/=" as the argument, DIRHISTORY will display a dialog to help you set the filename and command line options.

DO

If you specify "/[=]" for the ranges, DO will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, DO will display a dialog to help you set the individual attributes.

Sets three internal variables:

- %_do_dirs - The number of subdirectories traversed
- %_do_files - The number of files processed
- %_do_errors - The number of errors

EVENTLOG

If you pass "/=" as the argument, EVENTLOG will display a dialog to help you set the command line options.

FFIND

If you specify "/[=]" for the ranges, FFIND will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, FFIND will display a dialog to help you set the individual attributes.

Sets three internal variables:

- %_ffind_matches - The number of matches
- %_ffind_files - The number of files found
- %_ffind_errors - The number of errors

FOR

If you specify "/[=]" for the ranges, FOR will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, FOR will display a dialog to help you set the individual attributes.

Sets two internal variables:

- %_for_files - The number of files processed
- %_for_errors - The number of errors

FTYPE

/U - display/set the types in HKCU\Software\Classes

GLOBAL

If you pass "/=" as the argument, GLOBAL will display a dialog to help you set the command line options.

GZIP

If you specify "/[=]" for the ranges, GZIP will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, GZIP will display a dialog to help you set the individual attributes.

HEAD

If you pass "/=" as the argument, HEAD will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, HEAD will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, HEAD will display a dialog to help you set the individual attributes.

/B - ignore Bell (ASCII 7) characters.

Sets two internal variables:

%_head_files - The number of files displayed
%_head_errors - The number of errors

HISTORY

If you pass "/=" as the argument, HISTORY will display a dialog to help you set the command line options.

IFTP

If you pass "/=" as the argument, IFTP will display a dialog to help you set the command line options.

JABBER

If you pass "/=" as the argument, JABBER will display a dialog to help you set the command line options.

LIST

If you pass "/=" as the argument, LIST will display a dialog to help you set the file and command line options.

If you specify "/[=]" for the ranges, LIST will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, LIST will display a dialog to help you set the individual attributes.

MD

If you pass "/=" as the argument, MD will display a dialog to help you set the directory and command line options.

Sets two internal variables:

%_md_dirs - The number of directories created
%_md_errors - The number of errors

MKLINK

If you pass "/=" as the argument, MKLINK will display a dialog to help you set the file and command line options.

/A - create a link with an absolute path. (For CMD compatibility, MKLINK normally creates relative links if you don't pass the full pathname.)

/X - deletes a directory link.

Sets two internal variables:

- %_mklink_links - The number of links created
- %_mklink_errors - The number of errors

MKLNK

If you pass "/=" as the argument, MKLNK will display a dialog to help you set the file and command line options.

Sets two internal variables:

- %_mklnk_links - The number of links created
- %_mklnk_errors - The number of errors

MOVE

If you pass "/=" as the argument, MOVE will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, MOVE will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, MOVE will display a dialog to help you set the individual attributes.

Sets three internal variables:

- %_move_dirs - The number of directories created
- %_move_files - The number of files moved
- %_move_errors - The number of errors

PDIR

If you specify "/[=]" for the ranges, PDIR will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, PDIR will display a dialog to help you set the individual attributes.'

/B1 - display bare filenames with the relative path from the start, when used with /S.

Sets three internal variables:

- %_pdir_dirs - The number of directories displayed
- %_pdir_files - The number of files displayed
- %_pdir_errors - The number of errors

PLAYAVI

If you pass "/=" as the argument, PLAYAVI will display a dialog to help you set the filename and command line options.

PLAYSOUND

If you pass "/=" as the argument, PLAYSOUND will display a dialog to help you set the filename and command line options.

PLUGIN

If you pass "/=" as the argument, PLUGIN will display a dialog to help you set the file and command line options.

RD

If you pass "/=" as the argument, RD will display a dialog to help you set the directory and command line options.

If you specify "/[=]" for the ranges, RD will display a dialog to help you set the individual range arguments.

Sets two internal variables:

%_rd_dirs - The number of directories deleted

%_rd_errors - The number of errors

(Note that if you do an RD /S, the actual deletions are done by DEL, so check the DEL variables.)

REN

If you pass "/=" as the argument, REN will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, REN will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, REN will display a dialog to help you set the individual attributes.

Sets three internal variables:

%_ren_dirs - The number of directories renamed

%_ren_files - The number of files renamed

%_ren_errors - The number of errors

SELECT

If you pass "/=" as the argument, SELECT will display a dialog to help you set the command line options.

If you specify "/[=]" for the ranges, SELECT will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, SELECT will display a dialog to help you set the individual attributes.

Added support for @file lists.

SENDMAIL

If you pass "/=" as the argument, SENDMAIL will display a dialog to help you set the command line options.

/X - Send EHLO instead of HELO.

SET

/Q - don't echo result of /A when at the command line.

START

If you pass "/=" as the argument, START will display a dialog to help you set the command and options.

SWITCH

Added new command CASEALL, which should follow all the CASE statements but precede the DEFAULT. If a CASE statement has been executed, then CASEALL will also be executed; otherwise it is ignored.

CASE arguments can be literals, variables, or functions.

SYNC

If you pass "/=" as the argument, SYNC will display a dialog to help you set the directory and command line options.

If you specify "/[=]" for the ranges, SYNC will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, SYNC will display a dialog to help you set the individual attributes.

Sets three internal variables:

- %_sync_dirs - The number of directories created
- %_sync_files - The number of files copied
- %_sync_errors - The number of errors

TAIL

If you pass "/=" as the argument, TAIL will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, TAIL will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, TAIL will display a dialog to help you set the individual attributes.

/B - ignore Bell (ASCII 7) characters

Sets two internal variables:

- %_tail_files - The number of files displayed

%_tail_errors - The number of errors

TAR

If you specify "/[=]" for the ranges, TAR will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, TAR will display a dialog to help you set the individual attributes.

Sets two internal variables:

%_tar_files - The number of files compressed

%_tar_errors - The number of errors

TASKLIST

If you pass "/=" as the argument, TASKLIST will display a dialog to help you set the command line options.

TOUCH

If you pass "/=" as the argument, TOUCH will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, TOUCH will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, TOUCH will display a dialog to help you set the individual attributes.

Sets three internal variables:

%_touch_dirs - The number of directories touched

%_touch_files - The number of files touched

%_touch_errors - The number of errors

TREE

If you pass "/=" as the argument, TREE will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, TREE will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, TREE will display a dialog to help you set the individual attributes.

TYPE

If you pass "/=" as the argument, TYPE will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, TYPE will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, TYPE will display a dialog to help you set the individual attributes.

/B - ignore Bell (ASCII 7) characters.

Sets two internal variables:

- %_type_files - The number of files displayed
- %_type_errors - The number of errors

UNGZIP

If you specify "/[=]" for the ranges, UNGZIP will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, UNGZIP will display a dialog to help you set the individual attributes.

Added support for wildcards in the gzip filename.

Added range support.

UNTAR

If you pass "/=" as the argument, UNTAR will display a dialog to help you set the archive name and command line options.

If you specify "/[=]" for the ranges, UNTAR will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, UNTAR will display a dialog to help you set the individual attributes.

Added support for wildcards in the tar filename.

Added range support.

Sets two internal variables:

- %_untar_files - The number of files extracted
- %_untar_errors - The number of errors

UNZIP

If you pass "/=" as the argument, UNZIP will display a dialog to help you set the archive name and command line options.

If you specify "/[=]" for the ranges, UNZIP will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, UNZIP will display a dialog to help you set the individual attributes.

Added support for wildcards in the zip filename.

Added range support.

Sets two internal variables:

- %_unzip_files - The number of files extracted
- %_unzip_errors - The number of errors

ZIP

If you pass "/=" as the argument, ZIP will display a dialog to help you set the filename and command line options.

If you specify "/[=]" for the ranges, ZIP will display a dialog to help you set the individual range arguments.

If you specify "/A:=" for the attributes, ZIP will display a dialog to help you set the individual attributes.

Sets two internal variables:

- %_zip_files - The number of files compressed
- %_zip_errors - The number of errors

ZIPSFX

Added ranges support.

If you specify "/[=]" for the ranges, ZIPSFX will display a dialog to help you set the individual range arguments.

Sets two internal variables:

- %_zipsfx_files - The number of files compressed
- %_zipsfx_errors - The number of errors

Internal Variables:

TCEEXIT - the pathname of the TCEEXIT.xxx file, as of the time %_TCEEXIT is referenced. (The value could change before TCEEXIT.xxx is called.)

TCSTART - the pathname of the TCSTART.xxx file. It is set prior to TCSTART being executed.

Variable Functions:

- * The maximum argument size (passed to and returned from) variable functions has been increased to 32,767 characters.

%@ASSOC - added second optional argument "U" for associations in HKCU\Software\Classes.

%@B64DECODE - Decode a Base64 file or string (MIME encoding format).

%@B64ENCODE - Encode a file or string to Base64 (MIME encoding format).

%@FILEOPEN - added support for CON::

[%@FTYPE](#) - added second optional argument "U" for types in HKCU\Software\Classes.

[%@GETDIR](#) - now uses the newer Explorer-style dialog.

[%@HEXDECODE](#) - Decode a hex encoded file or string.

[%@HEXENCODE](#) - Encode a text file or string as hex.

[%@LINE](#) - is 50% faster.

[%@LINES](#) - is 50% faster.

[%@SELECT](#) - is 200% faster.

[%@UTF8DECODE](#) - Decode a UTF8 file or string.

[%@UTF8ENCODE](#) - Encode a UTF8 file or string.

[%@UUDECODE](#) - Decode a UU encoded file.

[%@UUENCODE](#) - Encode a file using the UU Encode format.

[%@YDECODE](#) - Decode a Y Encoded file

[%@YENCODE](#) - Encode file using the Y Encode format (similar to Base64, but uses 8-bit encoding to reduce the amount of data).

[What's New in Take Command 12](#)

[What's New in Take Command 11](#)

[What's New in Take Command 10](#)

2.4 Version 12

NEW VERSION OVERVIEW - Take Command 12.11

Feature List:

- * Added Spanish language support.

NEW VERSION OVERVIEW - Take Command 12.10

Feature List:

Added a Visual Studio 2010 theme to Take Command and the IDE/debugger.

- * Added Italian language support.
- * The non-English language support has been extended to include more menus, dialogs, and strings.

TCMD will now copy descriptions from DESCRIPT.ION when dragging & dropping in the Folders and List View windows.

TCMD and TCC will now display "Administrator:" on the title bar if the process is run as an administrator and no other title is defined.

Conditional tests (DO, IF, IFF, @IF) have two new tests:

string =~ expression Returns 1 if the string matches the regular expression

string !~ expression Returns 1 if the string does not match the regular expression

HTTP operations will now try to autodetect and use the system proxy settings, if available. (If not, they will fall back to using the values defined in OPTION / Internet.)

FTP / FTPS / SFTP operations will now try to autodetect and use the system firewall settings, if available. (If not, they will fall back to using the values defined in OPTION / Internet.)

Help file updates.

Commands:

CD

If the path argument is ~ (tilde), CD will change to the user's home directory (defined by HOME in the environment).

CDD

If the path argument is ~ (tilde), CDD will change to the user's home directory (defined by HOME in the environment).

GZIP

Fixed a (3rd party dll) bug when compressing with lzw.

FFIND

Added /8 option for UTF-8 files.

LIST

Added /8 option for UTF-8 files.

Variable Functions:

@HTMLDECODE - Decodes an HTML string.

@HTMLENCODE - Encodes the string for HTML (i.e., replacing things like > with >).

Batch Debugger:

Conditional breakpoints - you can now add breakpoint conditions to break on the number of iterations, or on a [conditional expression](#) (like the tests in IF and DO). There is a new tabbed window that shows the breakpoints and their (optional) conditions.

Added the option to disable (without deleting) breakpoints.

The breakpoints and conditions are saved when the batch file is saved & restored when the batch file is reloaded.

NEW VERSION OVERVIEW - Take Command 12.0

This is a summary of the compatibility fixes and new features. For complete details, see the appropriate topics in this help file.

The new features that are supported in **TCC/LE** are marked with a *.

Feature List:

- * Numerous optimizations to make everything a little faster.
- * Added Russian language support.
- * New icons for **Take Command**, **Take Command/LE**, **TCC**, and **TCC/LE**.

The **Take Command** folder view now auto updates when folders are created, deleted, or renamed. (This can be disabled with the new .INI directive AutoUpdateFolders.)

Added directory wildcard support to **TCC**. You can control the subdirectory recursion by specifying * or ** in the path. A * will match a single subdirectory level; a ** will match any all subdirectory levels for that pathname. Directory wildcards also support regular expressions. Directory wildcards cannot be used with the /O:... option (which sorts entries before executing the command). And think very carefully before using directory wildcards with a /S (recurse subdirectories) option, as this will almost certainly return unexpected results! There are a few commands which do not support directory wildcards, as they would be meaningless or destructive (for example, TREE, @FILEOPEN, @FILEDATE, etc.).

Take Command now supports customizing the color palettes used in the tab windows, via a "Tab Colors" button on the "Configure Take Command / Tab" dialog. **Take Command** will first try to retrieve the palette from the console (Vista and later only). If the console is using a custom palette, **Take Command** will use that palette for the tab window. If there is no custom palette for this console, **Take Command** will use the colors saved from the "Tab Colors" dialog.

The **Take Command** "Configure Take Command / Tabs / Windows" foreground and background colors combo boxes now display the actual colors instead of the color name. (This is necessary to support the custom color palettes, as otherwise if you redefined the palette the color names would have no relation to the colors used. It also makes it easier to select the color you want.)

You can now use environment variables in the TCMD.INI sections "[4NT]" and "[TCMD]", and the ini parser will expand them before setting the directives. (Note that if you were already setting a directive to a value with an embedded %, you'll need to double the %'s now.)

- * TCMD.INI file parsing (and thus the startup time) is faster.

The **TCC** persistent History and DirHistory lists load significantly faster.

Changed the internal **TCC** Perl support from the hopelessly buggy & undocumented embedded Perl to PerlScript (the WSH COM interface to Perl). The good news is that Perl support is no longer version dependent, so you can use Perl 5.8, 5.10, or 5.12.

TCC now supports Python 3.1, 2.6, and 2.5. (**TCC** will search for the Python dll's in that order.)

The **Take Command** tabbed toolbar will hide the tabs row if there is only one tab defined.

The default directory for the Command Input window is now set to the selected directory in the Folders view. (This allows tab completion in the Command Input window to match the List View when no path is specified.)

The **Take Command** "Tabs" menu now includes "Previous Tab" and "Next Tab", allowing you to define keys to change tabs from the keyboard.

The **Take Command** "Options" menu now includes "Configure Tabbed Toolbar", which allows you to add buttons or tabs, or save or reload the toolbar from TCMD.INI. (You can also do this by right clicking on the toolbar.) To edit an existing button, right click on the button.

Ranges now support multiple file exclusions (i.e., "dir /[!*.txt] /[!*.doc] *"). (Useful for aliases or variable substitution where a default file exclusion is specified.)

Added "Owner" to file ranges. The syntax is "/[O"owner"]" or "/![O"owner"]". It supports wildcard comparisons; the value returned is the same as shown in DIR /Q or [%@owner\[file\]](#).

If you left click on the current **TCC** command line while in a **Take Command** tab window, **TCC** will move the cursor position to the mouse position.

You can describe directories and files with the List View in **Take Command** when it is in "Details" view. Double click on the "Description" column, enter or edit the description in the edit box, and press Enter to save.

The toolbar button dialog now has a "Copy" button, which copies the properties of the current button to a new button.

- * Changed the dialog fonts from MS Sans Serif to Tahoma.

The VK_APPS (menu) key will invoke the context menu in **Take Command** tab windows.

The **Take Command** and **TCC** configuration dialogs have been reformatted to display correctly on Netbook (1024x600) displays.

Added (experimental) double-wide character set (i.e., Japanese, Chinese, etc.) support to **Take Command**.

- * File read operations are slightly faster.

Startup Options:

- /C Run the specified command in a new **TCC** tab window. If there is already a **Take Command** session running, /C creates a new tab in the existing **Take Command** rather than starting a new session. /C must be the last option on the command line (otherwise **Take Command** can't tell if additional options belong to **Take Command** or the command to run in the **TCC** tab).
- /NT Don't load the default startup tabs (usually only useful when combined with /C or /T).

.INI Directives:

AutoUpdateFolders=yes|NO - Updates the **Take Command** folder view automatically when folders are created, deleted, or renamed. This will slow things down slightly, so if you have processes that are modifying directories a lot (more than once every few seconds) you may want to disable the autoupdates.

- * CMDVariables=yes|NO - Requires environment variables in **TCC** to be delineated by leading and trailing %'s. For those of you who just can't get enough of CMD compatibility, and the consistent way it handles variables. Which have to be enclosed in %'s, like "%varname%". Unless it's a batch variable, which only has a leading %. Or a FOR variable, which only has a leading %. Or sometimes two leading %'s.

CommandInputWrap=YES|no - If set to Yes, **Take Command** will wrap the input lines in the Command Input window instead of scrolling horizontally.

CompletePercents=yes|NO - If the tab-completed filename has embedded %'s, and the first argument of the command is an internal command, the %'s will be doubled so that variable expansion won't end up deleting (or unexpectedly expanding) the filename. (The tab completion routine cannot preparse aliases or environment variables, so if you're using one of those as your first argument, you're out of luck!)

DirHistoryOnEntry=yes|NO - **TCC** normally saves the previous directory to the directory history when you change to a new directory. This option saves the new directory to the directory history when you change directories.

IBeamCaret=yes|NO - use an IBeam caret instead of the default console underline caret. (**Take Command** tab windows only.)

LockListView=yes|NO - Prevents modification of the **Take Command** List View (double clicking on the description column to edit the description, or selecting a file or directory and pressing F2 to rename it).

SmoothScroll=yes|NO - Scrolls the **Take Command** tab windows smoothly (1 pixel at a time instead of a line at a time). Slows things down and doesn't provide any extra functionality; it's just eye candy! (In order to reduce the slowdown, **Take Command** will only use smooth scrolling when the console has scrolled <= 3 lines. If the console is spitting out a lot of short lines, you won't see the smooth scroll, but then you can't really discern anything at that speed.)

StartTabWait - the number of milliseconds to wait before loading the next **Take Command** startup tab. The range is 0 - 5000. (This should only be needed in rare cases, when tabs are interfering with one another while starting.)

Command Line Editing:

See the new .INI directive CompletePercents.

You can triple-click on a line to select the entire line (**Take Command** "Command Input" and tab windows only).

Command completion (i.e., Tab or F9 for the first argument on the line) will append a space to the command name.

^P in the Command Input window will print the command history.

^P (last argument in previous command) on the **TCC** command line has been changed to **^B**. (But it can still be remapped.)

New Commands:

CLIPMONITOR

Monitor the Windows clipboard, and execute a command when the contents change.

CLIPMONITOR [/C]
CLIPMONITOR *n* *command*

n Number of repetitions (or **FOREVER**)
command Command to execute when condition is triggered

/C Remove the clipboard monitor.

FONT

Change the console font characteristics. This command is only available in Windows Vista and later, and will only affect stand-alone **TCC** console windows. (You can already change the font in **Take Command** tab windows.) The syntax is:

FONT [/F*family* /N*name* /W*n* /X*n* /Y*n*]

/F - The font family:
decorative
dontcare
modern
roman
script
swiss

/N - Font face name

/W - Font weight (100 - 1000, in multiples of 100). The normal weight is 400; bold is 700.

/X - The width of each character, in logical units.

/Y - The height of each character, in logical units.

GZIP

The GZIP command is compatible with the archives created by the UNIX gzip utility and supports RFC 1952. GZIP is used for compressing a single file; if you need to compress multiple files you should use the ZIP (or TAR) command. The syntax is:

GZIP [A:[[-][+]rhsdaecjot] /A /E /L /M /O:[-]adegnrstu /Q /V] **gziparchive** [@file] file

gziparchive The .gz file to work with

path The path where the file will be extracted

file The file to extract

/A:... - Attribute selection

/A - Add file (default).

/E - Compression method (0 = deflate, 1 = lzw). The default is 0.

/L - Compression level (1 - 6). The default is 4.

/M - Move the file to the gzip archive and delete the original on disk

/O - Sort order

/Q - Quiet (don't display filenames as they are added to the archive).

/V - View date, time, and filename of the file in the archive. (Due to the limitations in the GZIP format, this can only display the first file in the archive.)

TAR

The TAR command is compatible with archives created by the UNIX tar utility. TAR also supports gzip compression and can be used to create .tar.gz archives.

TAR [A:[[-][+]rhsdaecjot] /A /C /D /F /G /M /O:[-]adegnrstu /P /Q /R /U /V] **tararchive** [@file] file...

tararchive The tar file to work with

file The files(s) to be added to the zip file

/A:... - Attribute selection

/A - Add files to the tar archive (default).

/C - Display contents of the tar archive.

/D - Delete the specified file(s) from the tar archive.

/F - Update only those files that currently exist in the tar archive, and which are older than the files on disk.

/G - Use gzip compression.

/M - Move

/O - Sort order

/P - Display progress (in %) for each file.

/Q - Quiet (don't display filenames as they are added to the archive).

/R - If the argument is a subdirectory, copy all of the files in that subdirectory and all of its subdirectories to the tar archive.

/U - Update which either don't exist in the tar archive, or which are older than the files on disk.

/V - View the list of files in the tar archive (date, time, size, and filename).

UNGZIP

The UNGZIP command is compatible with the UNIX gunzip utility and supports RTF 1952. The syntax is:

GZIP [/E /O /Q /V] *gziparchive* [*path*]

gziparchive The .gz file to work with

path The path where files will be extracted

/E - Extract file (default).

/O - Overwrite existing file.

/Q - Quiet (don't display filenames as they are extracted from the archive).

/V - View date, time, and filename of the file in the archive. (Due to the limitations in the GZIP format, this can only display the first file in the archive.)

UNTAR

The UNTAR command decompresses archives created by the TAR command. UNTAR also supports gzip decompression and can be used to extract .tar.gz archives.

UNTAR [/C /D /E /F /G /Nte /O /Q /U /V] *tararchive* *path* ...

tararchive The Tar file to work with

path The path where files will be extracted

file The file(s) to extract

/C - Display contents of the tar archive.

/D - Recreate the directory structure saved in the tar file.

/E - Extract (default)

/F - Extract only those files that currently exist in the target folder, and which are older than the file in the tar archive.

/G - Gzip

/N[te] - Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*).

/O - Overwrite existing files.

/P - Display progress (in %) for each file.

/Q - Quiet (don't display filenames as they are extracted from the archive).

/U - Extract files which either don't exist in the target folder, or which are older than the file in the zip archive.

/V - View the list of files in the archive (date, time, size, and filename)

ZIPSFX

ZipSFX creates a zip-compatible self-extracting archive. The syntax is:

ZipSFX [/B"text" /C"text" /D"path" /F"file" /Ln /M"message" /R /S"password" /X64] archive directory

archive The name of the self-extracting executable
directory The directory to be compressed into the self-extracting executable

/B - Banner text to display before the self-extraction begins.

/C - Caption for the self-extractor dialogs.

/D - Target directory for the self-extractor.

/F - Optional name of the file to execute (open) after the archive is extracted. This must be a relative path to a file in **directory**. If this is set to ".", the folder in which the archive has been decompressed will open in Windows Explorer. If it is set to "" (empty string), the extractor will close and take no action.

/Ln - Compression level (0 - 6; the default is 4).

/M - Message to notify the user that the extraction has completed normally.

/R - Recurse subdirectories

/S - Password

/X64 - Create a 64-bit executable.

Commands:

ACTIVATE

ICON=file - Change a window's caption bar and task bar icon. The file can be an icon file or an executable. (If an executable, ACTIVATE will use the first icon.)

ALIAS

/O - Don't overwrite existing value (only valid in combination with /R)

DEL

/W now uses DoD 5220.22-M (E) for secure deletion.

DIR

/G:nn - Set the disk cluster size to be used by /G. DIR will normally query the system for the cluster size on the specified drive, but you can override with /G:n if you know that the returned info is incorrect, or if you want to find the size required if the specified files were moved to another device with a different cluster size.

/NL - Don't display the link name for symbolic links.

DO

Now supports usage in aliases and on the command line. You need to enclose the body of the DO loop in a command group that follows the DO expression, and there is no ENDDO. The syntax is:

```
DO [n | FOREVER] (commands)
    or
DO varname = start TO end [BY n ] (commands)
    or
DO FOR n SECONDS | MINUTES | HOURS (commands)
    or
DO [WHILE | UNTIL [DATETIME yyyy-mm-dd hh:mm:ss]] condition (commands)
    or
DO varname IN [range] [/I"text"] [/D"path" /N[dj] /O:[-]adegnrstu /Sn] [/A:[-][+]rhsdaecjot] [/C
/L /P /T"...""] [@]set (commands)
```

For example:

```
DO count = 1 to 10 (echo count = %count)
```

ITERATE and LEAVE are supported in command-line DO's.

ISHUNG is a new condition (for DO WHILE and DO UNTIL) that is true if the specified window is not responding.

ENDLOCAL

Restores the Function list.

FFIND

/Ln - The number of leading and trailing lines to display on a match. Each successive group of lines in a file will be separated by a "----" header.

/N[dehjs]

D - don't search hidden directories
E - don't display errors
H - no header
J - don't search junctions or symlinks
S - no footer

FOR

/NJ - Don't recurse into search junctions or symlinks

/W - Indicates that the FOR set is to be treated as filenames, even if no wildcards are detected (or if you're using regular expressions).

FUNCTION

/O - Don't overwrite existing value (only valid in combination with /R)

HISTORY

/F"..." - Delete matching command lines. You can have multiple /F"..." arguments, and they can contain wildcards.

/R1 - Ignore duplicates and HistoryExclude and always append the lines. (This is considerably faster for large history lists.)

IF

ISHUNG is a new condition that is true if the specified window is not responding.

IFTP

/R - Automatically reconnect if the ftp session times out.

JABBER

Now supports SSL, so it will work with SSL servers like Google Talk.

/Tn - Set the port number (default is 5222).

/V - Display verbose output (useful for debugging).

LIST

Added horizontal mouse wheel scrolling in **Take Command** tab windows (requires Vista or later).

MD

/D will change to the newly created directory.

MOVE

Now displays the empty source directories it is removing (with /S and no /Q).

/K - If the move was to a different drive, this option will move the original to the recycle bin instead of deleting it.

/Ns - Don't display the summary of files moved.

/W now implements DoD 5220.22-M (E) for secure deletion.

ON

DBLCLICK [command] - Execute the specified command on a left mouse button double click.

OPTION

There is a new "Console Palette" button on the Windows tab that allows you to redefine the 16 console color attributes. (Requires Vista or later.) This will *not* work when running **TCC** in a **Take Command** tab windows due to a bug in the Windows API (it insists on unhiding hidden console windows), but when in **Take Command** you can define a custom palette for the tab windows.

The "Windows" foreground and background colors combo boxes now display the actual colors instead of the color name. (This is necessary to support the custom color palettes, as otherwise if you redefined the palette the color names would have no relation to the colors used. It also makes it easier to select the color you want.)

QUERYBOX

/CUE="text" - Displays the cue text in light gray in the edit box (it disappears as soon as you enter a character).

SELECT

/O: - Added /O:o (sort by owner) option

/Q - Display owner (you'd better have a > 80 column display!)

SENDMAIL

/SSL[=n] - Use SSL to connect to the server. Supports automatic, implicit, and explicit modes.

SET

/O - Don't overwrite existing value (only valid in combination with /R)

SETLOCAL

Now saves the Function list as well as the environment & aliases

START

/ELEVATED - Start the program with full admin privileges. (Windows Vista or later only.)

TASKBAR

LOCKDESKTOP - Locks the desktop (Vista and later)

LOGOFF - Display the log off dialog

USER - Display the switch user dialog

TASKLIST

/D - Show the modules loaded in each process

/M - Show the memory usage for each process

/N - Show the class names

/T - Show the kernel and user cpu times

UNZIP

Is now using a different Zip library.

/CRC - Display the file CRCs (only when using /V).

/I - Save the compressed file's description ("File Comment") to the NTFS description or DESCRIPT.ION file.

/P - Display progress (in %) for each file.

ZIP

Is now using a different Zip library.

/CRC - Display the file CRCs (only when using /V).

/En - set the encryption level (0=default, 1=AES 128-bit, 2=AES 192-bit, 3=AES 256-bit)

/I - Save the file's description (from the NTFS description or DESCRIPT.ION) as the compressed file's "File Comment"

/Ln - set the compression level (0-6, where 0 = no compression and 6 = maximum compression). The default is 4.

/P - Display progress (in %) for each file.

/T - Save the file attributes (they will be written to the file system during extraction)

/Z"..." - set the comment for the zip file.

Internal Variables:

EXECARRAY - The number of array elements assigned by the last @EXECARRAY function.

[INSERT](#) - The current insert edit mode (0=overstrike, 1=insert)

[XMOUSE](#) - The column position of the most recent left mouse click. (Note that this will only work in a **Take Command** tab window, or if you have enabled the console mouse in a stand-alone **TCC** session.)

[YMOUSE](#) - The row position of the most recent left mouse click. (Note that this will only work in a **Take Command** tab window, or if you have enabled the console mouse in a stand-alone **TCC** session.)

Variable Functions:

[@EVAL](#) - The <<, >>, MOD, and \ operators now support large (up to 10,000 digit) numbers. (Previously they were "limited" to 64-bit integers). The bitwise operators (AND, OR, XOR) are still limited to 64-bit integers.

[@EVAL](#) - Now supports binary input by prefixing "0b" to the number:

(%@eval[0b1001+2]

[@EVAL](#) - Now supports binary output by appending "=b" to the expression:

(%@eval[1+1=b]

[@GETDATE](#)[date] - Displays a calendar dialog and returns the selected date (yyyy-mm-dd). You can optionally pass the default date.

[@GETDATETIME](#)[datetime] - Displays a date and time picker and returns the selected date and time (yyyy-mm-dd hh:mm:ss). You can optionally pass the default date and time.

[@INIWRITE](#) - If you don't specify an entry, will now delete the specified section.

[@PPID](#)[file] - Returns the PID for the parent process of the specified executable

[@REGBREAD](#)[key,handle,length] - Read a registry value into a (previously created) binary buffer

[@REGBWRITE](#)[key,type,handle,length] - Write a registry value from a binary buffer

[@REGEX](#) - Removed the (unused and confusing) group count return value. @REGEX now only returns 1 (for a match) or 0 (for no match).

[@REGSET](#) - Added support for REG_MULTI_SZ. (The data values are separated by commas.)

[@TARCOUNT](#)[file] - Return the number of files in a .tar archive

[@TARCFILE](#)[file,i] - Return the compressed name of a file in a .tar archive

[@TARDFILE](#)[file,i] - Return the decompressed name of a file in a .tar archive

[@TARFILEDATE](#)[file,i]- Return the date & time of a file in a .tar archive

[@TARFILESIZE](#)[file,i]- Return the size of a file in a .tar archive

[@URLDECODE](#)[string] - Decode an URL encoded string (replacing %xx with the original characters)

[@URLENCODE](#)[string] - Encode a string for transmission (replacing non-alphanumeric characters with their %xx hex representation)

[@ZIPCOUNT](#)[file]- Return the number of files in a .zip archive

[@ZIPCOMMENT](#)[file] - Return the comment for the .zip archive

[@ZIPCFILE](#)[file,i] - Return the compressed name of a file in a .zip archive

[@ZIPDFILE](#)[file,i] - Return the decompressed name of a file in a .zip archive

[@ZIPFILECOMMENT](#)[file,i] - Return the comment for a file in a .zip archive

[@ZIPFILECRC](#)[file, i] - Return the CRC for a file in a .zip archive

[@ZIPFILEDATE](#)[file,i] - Return the date & time of a file in a .zip archive

[@ZIPCFILESIZE](#)[file,i] - Return the compressed size of a file in a .zip archive

[@ZIPDFILESIZE](#)[file,i] - Return the decompressed size of a file in a .zip archive

Batch Debugger:

The batch debugger has been rewritten with a new editor and many new features.

If you hover the mouse over a variable, the debugger will display a tooltip with the current value.

If you hover the mouse over an internal **TCC** command, the debugger will display a tooltip with the command syntax.

If you hover the mouse over an array variable (1-dimensional only!), the debugger will display a tooltip with up to the first 20 elements with assigned values.

The "Aliases" and "Functions" windows now support syntax coloring.

Added "Add to Watch" to the context menu.

The Goto dialog now has an optional Column position.

The File menu has a new option:

Save Copy As - Saves a copy of the file to a new name, without changing the default name.

The Edit menu has new options:

Move Line Up - Moves the current line up one row.

Move Line Down - Moves the current line down one row.

Toggle Comment - Inserts / Removes a "rem " at the beginning of the current line.

Remove Blank Lines - Removes blank lines from the selection (or the entire file if no selection).

Compress Spaces - Removes extra spaces between words for the selection (or the entire file if no selection).

Make Selection Uppercase

Make Selection Lowercase

View Whitespace - Displays a marker (a small dot) in the columns for spaces or tabs.

View EOL - Display the end of line characters (CR and/or LF).

The Debug menu has a new option:

Pause On Error - Switches the debugger to single step mode when **TCC** encounters an error.

The Options menu entry has new options:

Tabs - Change the tab and indent settings.

Display Line Numbers - Toggles the line numbering on & off.

Display Folding Margin - Toggles the folding margin (the + indicator) on and off.

Indentation Guides - Prints vertical lines at the current indent columns (useful for lining up code).

The Windows menu entry has new options:

Zoom In - Increase text size by one point.

Zoom Out - Decrease text size by one point.

Reset Zoom - Reset the text size to the original size.

Plugins:

Added UNKNOWN_CMD, PRE_INPUT, PRE_EXEC, and POST_EXEC support. **TCC** will first look for aliases of those names; if it doesn't find a match it will look in the plugins for a matching name. For UNKNOWN_CMD and PRE_EXEC, **TCC** will pass the command line to the plugin; PRE_INPUT and POST_EXEC will get a NULL.

2.5 Version 11

NEW VERSION OVERVIEW - Take Command 11.0

This is a summary of the compatibility fixes and new features. For complete details, see the appropriate topics in this help file.

The new features that are supported in TCC/LE (including the TCC/LE component of Take Command/LE) are marked with a *.

Feature List:

Both Take Command and TCC are now also available in x64 versions.

You can now put (simple) GUI apps into tabs. Note that this will not work for apps that have multiple parent windows!

The Take Command toolbar is now a "tabbed toolbar", allowing up to 20 tabs and 50 buttons per tab. You can also now right click on any button to edit it.

Take Command supports remapped console color palettes in the tab windows. If you are running Vista or later, Take Command will use the individual palette defined for each console.

Display output in Take Command tab windows is 20% faster than v10.

Take Command can now optionally automatically attach all console apps to tabs, regardless of how or when they are started.

The "Attach Tabs" menu option in TCMD now includes hidden console windows. (This allows you to reattach consoles that may have been orphaned from a TCMD crash or unusual shutdown.)

You can now drag and drop into the Folders View.

The popup windows (history, file searching, etc.) now display the current search string on the left side of the window toolbar.

TCC now loads more than twice as fast as in v10.

Added SSH FTP (SFTP) support.

The FTP, FTPS, and SFTP syntax now accepts a "*" as the password as a request for an interactive password prompt; i.e.:

```
dir ftps://bob:*@ftp.jpsoft.com/mydir
```

The new environment variable PROMPT2 defines the prompt used for line continuations (i.e., when the last character on the line is an escape character). The default is "More? ".

- * The directory stack (DIRS, PUSHD, POPD) size has been increased to 4K.

Added embedded Tcl / tk support.

The TCMD tab labels can be rotated 90 degrees (see TabRotation in the .INI directives), allowing you to fit a lot more labels at the cost of slightly smaller tab windows

- * Added support for embedded variables in the CMD delayed expansion (!var!) syntax.

The Command Input window can now be autosaved & autoloaded. (See the CommandInputFile .INI directive below.)

- * DESCRIPT.ION reads are 500% faster.

Increased the maximum number of Take Command startup tabs to 25.

Increased the maximum argument size in TCC to 8191 characters.

Added a "Run..." dialog option to the Take Command tab context menu.

The file processing commands (COPY, DEL,DO, FOR, MOVE, RENAME, etc.) have a new /O:... option to sort the files before they are processed. The sorted filenames are saved to memory before being passed to the command; this allows you to dispense with temporary files when the command might otherwise process the same filename twice (for example, with FOR and RENAME).

Startup Options:

TCC /IX - don't execute TC EXIT

TCC /Q - don't display copyright / version message (registered copies only)

.INI Directives:

This list is for your information only. You should always use OPTION (in TCC) or "Options / Configure Take Command" (in Take Command) to set your TCMD.INI options.

AutoAttachConsole - if Yes, TCMD will periodically look for and create a new tab for any unattached console windows. (Note that this means you cannot ever detach a console tab!)

AutoCDD - if No, disables the automatic directory changes (i.e., directory name with a trailing \) as the only argument on the command line

ClosePrompt - if 1, Take Command will pop up a message box to confirm exiting

CommandInputFile - name of a file used to save & restore the Command Input window

HistCase - if Yes, command history comparisons will be case sensitive

LockExplorerBar - if 1, TCMD will lock the Explorer toolbar in place (so it cannot be moved or docked)

LockMenuBar - if 1, TCMD will lock the menu bar in place (so it cannot be moved or docked)

LockTabBar - if 1, TCMD will lock the tab toolbar in place (so it cannot be moved or docked)

NoINIErrors - if Yes, suppresses error messages when parsing TCMD.INI

SSHLLocalPort - the TCP port in the local host where IPPort binds.

SSHLLocalHost - the name of the local host or user-assigned IP interface through which connections are initiated or accepted.

SSHPort - the port on the SSH server where the SSH service is running (default is 22).

TabRotation - if 1, TCMD will rotate the tab labels (and text) 90 degrees. (This allows you to fit many more tabs in the window, at the cost of a reduced window size.)

Tcl - if Yes, TCC will execute *.tcl scripts.

Command Line Editing:

Tab completion now supports internal variables

Tab completion now supports variable functions

Tab completion now checks for ftp / ftps names (<ftp://xxx>) and won't break on the first /

New Commands:

UNZIP - Extract files from a zip archive. UNZIP will automatically use the Zip64 extensions if the archive is in Zip64 format. The UNZIP syntax is:

UNZIP [/C /D /E /F /O /S"password" /U /V] ziparchive path ...

VBEEP - flash the screen (by setting all the attributes to their inverse) and optionally beep the speaker. The syntax is the same as BEEP:

VBEEP [frequency duration...][asterisk | exclamation | hand | question | ok]

ZIP - Add, update, or delete files to a zip archive. UNZIP will automatically use the Zip64 extensions if the archive is in Zip64 format. The ZIP syntax is:

ZIP [A:[[-][+]rhsdaecjot] /A /C /D /F /M /O:[-]adegnrstu /P /Q /R /S"password" /U /V /YC]
ziparchive [@file] file...

Commands:**ATTRIB**

/O:xxx to sort files before they are processed

+C | -C - compress or uncompress the file or directory

BEEP

Now supported in the x64 version of TCC. (Because 64-bit versions of Windows do not support playing sounds through the Windows Beep API, TCC x64 uses DirectSound for BEEP.)

CDD

/Un sets recursion depth for JPSTREE.IDX updates (like /Sn)

COPY

Added number of files that failed to copy to the result

/O:xxx to sort files before they are processed

/Nn will not update the file descriptions (either in DESCRIPT.ION or an NTFS stream)

/W will delete files in the target directory that don't exist in the source directory (use this instead

of SYNC when you only want to synchronize "one-way")

DEL

/O:xxx to sort files before they are processed

Added number of files that failed to be deleted to the result

/S /X displays the directories removed (with a trailing \)

/L deletes symlinks instead of their contents

/Nn will not update the file descriptions (either in DESCRIPT.ION or an NTFS stream)

DESCRIBE

/O:xxx to sort files before they are processed

DIR

/F now supports colorization

/B /S now supports colorization

DIRS

+n / -n - rotate the directory stack up or down by the specified amount

/Q - don't display the directory stack (only useful when combined with +n or -n)

DO

/O:xxx to sort files before they are processed

ENDLOCAL

Can now be used at the command line (including aliases). The maximum nesting level is 10.

FFIND

/H will skip binary files (user-configurable file extensions) when searching.

FOR

/O:xxx to sort files before they are processed

HEAD

/N+n - skip first n lines

/O:xxx to sort files before they are processed

HISTORY

/R - if you load a file that is larger than the history list size, HISTORY will only load the last part of the file that will fit

/Tn - if n is positive, only display the last 'n' history entries. If n is negative, skip the first 'n' entries.

/V - Display the history in reverse order (most recent first)

INPUT

/K"..." - only accept the specified characters

LIST

/O:xxx to sort files before they are processed

MD

/C - create a compressed directory

MOVE

/O:xxx to sort files before they are processed.

Added number of files that failed to be moved to the result

/Nn will not update the file descriptions (either in DESCRIPT.ION or an NTFS stream)

/Ns will not display the summary

OPTION

//directive with no value will reset to the default value

/U - check <http://jpsoft.com> for updates

PDIR

Added support for escaped characters in separator text

PLUGIN

/C - only display commands

/F - only display variable functions

/K - only display keystroke plugins

/V - only display internal variables

REN

/O:xxx to sort files before they are processed

Added number of files that failed to be renamed to the result

/Ns will suppress the summary

/Nn will not update the file descriptions (either in DESCRIPT.ION or an NTFS stream)

SETLOCAL

Can now be used at the command line (including aliases). The maximum nesting level is 10.

SYNC

/O:xxx to sort files before they are processed

/Nn will not update the file descriptions (either in DESCRIPT.ION or an NTFS stream)

TAIL

/O:xxx to sort files before they are processed

TASKLIST

/C will display the current priority class for each process

/L will display the startup command line for the process (this replaces the window title in the output)

TCTOOLBAR

Added a new parameter at the beginning of the argument list to specify on which tab the button should be set. (The tab to use is specified by its label.)

TOUCH

/O:xxx to sort files before they are processed

/R can now copy an existing directory's timestamp to a newly created (/C) file

TREE

/O:xxx to sort files before they are processed

/Z without /F will now display the directory tree sizes. (Each directory size is the size of the current directory and all of its subdirectories.)

TYPE

/O:xxx to sort files before they are processed

/X supports binary files

WHICH

Plugin commands now show the plugin name (i.e., "Foo is a plugin command (Foobar)")

WINDOW

DETACH - detach the TCC process from a TCMD tab window

Internal Variables:

_CONSOLEB - the handle to the console screen buffer

_ISODOWI - ISO 8601 numeric day of week (Mon=1, Sun=7)

_ISOWDATE - ISO 8601 current week date (yyyy-Www-d)

_ISOWEEK - ISO 8601 week of year

_ISOWYEAR - ISO 8601 week date year

_SERVICE - returns 1 if TCC was started as a service (TCC /N)

_TCTABACTIVE - returns 1 if this TCC instance is the active tab in Take Command

_WOW64DIR - returns the system Wow64 directory (x64 Windows only)

_X64 - returns 1 if TCC is the x64 (64-bit) version

Variable Functions:

@AGEDATE - added support for ISO 8601 formats 5 (yyyy-Www-d) and 6 (yyyy-ddd)

@BPEEK, @BPEEKSTR, @BPOKE, @BPOKESTR, @BREAD, @BWRITE - now accept either decimal or hex arguments for offset / size / length

@CONSOLEB[handle] - create or restore a console screen buffer. "Handle" is the handle to the desired screen buffer. If "handle" is -1, @CONSOLEB just returns the current buffer handle. If "handle" is 0, @CONSOLEB will create and activate a new console screen buffer. If "handle" is non-zero, @CONSOLEB will switch to that screen buffer. @CONSOLEB returns the handle to the active screen buffer. @CONSOLEB allows you to preserve the contents of the current screen buffer by switching to a second buffer temporarily and then back to the original buffer.

@DATECONV[date,format] - convert date from one format to another format (output):

- 0 system default
- 1 USA (mm/dd/yy)
- 2 European (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO 8601 (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

@DISKFREE, @DISKUSED, @DISKTOTAL, @DOSMEM, @WINMEMORY - added E/e (Exabytes) option. (Zettabytes and Yottabytes will have to wait for 128-bit CPUs and OS's.)

@EXECSTR - now supports a negative start line number to mean "starting at the end and counting back"

@EXPAND - added support for exclusion ranges

@FILESIZE - added support for Terabytes (t/T), Petabytes (p/P) and Exabytes (e/E)

@ISODOWI[date] - ISO 8601 numeric day of week

@ISOWEEK[date] - ISO 8601 numeric week of year

@ISOWYEAR[date] - ISO 8601 numeric week date year

@ISPRIME[n] - returns 1 if the (64-bit) n is a prime number

@MAKEAGE - added support for ISO 8601 yyyy-Www-d and yyyy-ddd dates

@MAKEDATE - added support for ISO 8601 formats 5 (yyyy-Www-d) and 6 (yyyy-ddd)

@PIDCOMMAND[pid] - return the startup command line for the specified process ID

@PLUGINVER[plugin] - returns the version number (major.minor.build) for the specified plugin

@PRIME[n] - returns the first prime >= (64-bit) n

@PRIORITY[pid] - return the priority class for the specified PID. The return values are (in hex):

8000 - Above normal

4000 - Below normal

100 - realtime

80 - High

40 - Idle

20 - Normal

@PROCESSTIME[pid,n] - return the process times as a fileage. n is the time to return:

0 - start time

1 - end time

2 - kernel mode time

3 - user mode time

@SYSTEMTIME[n] - return the system times as a fileage. n is the time to return:

0 - idle time

1 - kernel mode time

2 - user mode time

@TCL - execute a Tcl command

@TK - execute a Tk script

Batch Debugger:

The evaluation window (Alt-F11) now supports copying the result to the clipboard

If you right click in the first column of the Watch window, the debugger will pop up an environment variable listbox. If you select an entry, it will be added to the watch list.

Plugins:

Updated the TakeCmd.h file with the new functions

Added some new functions for manipulating the directory history and command history:

DirHistoryStart(void) - returns a pointer to the beginning of the directory history

HistoryStart(void) - returns a pointer to the beginning of the command history

DeleteFromHistory(LPTSTR lpszLine) - deletes the line from the command history (this is a pointer to the line to be deleted, not a line to be matched!)

If the user tries to display online help with HELP, F1 or Ctrl-F1, TCC will check for a plugin variable, variable function, or command, and if the name matches search for, load and execute a "Help" function in the plugin. The plugin is responsible for displaying its own help. The "Help" function should NOT appear in the plugin's comma-delimited function list in pszFunctions. Help should return 1 if it displayed help (or if it doesn't want TCC to try to display help for this topic). The syntax of the Help function in the plugin should be:

```
Help( LPTSTR pszName );
```

If Take Command wants to display usage text, TCC will check for a plugin command, and if the name matches search for, load and execute a "Usage" function in the plugin. The plugin is responsible for displaying its own help. The "Usage" function should NOT appear in the plugin's comma-delimited function list in pszFunctions. The plugin should return a multi-line string containing the command syntax. The first line (terminated by a \r) is displayed in the Take Command status bar. The entire string is displayed as a tooltip popup when the mouse hovers over the status bar message. Usage should return 1 if it wrote something to pszUsage (or if it doesn't want TCC to try to display a usage string). The syntax of the Usage function in the plugin should be:

```
Usage( LPTSTR pszName, LPTSTR pszUsage );
```

The TakeCommandIPC function allows plugins to communicate with the controlling Take Command instance. The syntax is:

```
DLLExports int TakeCommandIPC( LPTSTR pszCommand, LPTSTR pszArguments );
```

The supported commands are:

HWND

Returns the Take Command window handle in pszArguments

TCTAB

Returns 1 if the process ID in pszArguments is running in a TC window

TCTABS

Returns the number of Take Command tab windows

HVIEW

Returns the handle of the active tab window in pszArguments

HELP

Displays the Take Command help for the topic in pszArguments

USAGE

Display the usage message in pszArguments in the status bar. The first line (up to the first CR) is displayed in the status bar; the rest is displayed in the tooltip if you hover the mouse over the status bar.

STATUSBAR

Display the message in pszArguments in the status bar

TCTOOLBAR

Update the Take Command tab toolbar with pszArguments

TCFILTER

Return the selected filter in the list view in pszArguments

TCFILTER_CMD

Set the selected filter in the list view to the value in pszArguments

CDD

Change the folder and list view to the directory in pszArguments

TCFOLDER

Return the selected folder in the Folders tree control in pszArguments

SHORTCUT

Return the name of the shortcut that started Take Command in pszArguments

SELECTED

Return the currently selected text in pszArguments

SELECT

Mark the selection specified in pszArguments (top, left, bottom, right) in Take Command

START

Attach a hidden console window whose hex PID is in pszArguments

ACTIVATE

Activate the window whose handle is in pszArguments

WINDOW

Has a number of arguments (specified in pszArguments) to control the Take Command window:

MAX

MIN

HIDE

RES

TRAY

TRANS=n

FLASH=n

DETACH n (where n is the PID of the process to detach)

TOPMOST

NOTOPMOST
TOP
BOTTOM

2.6 Version 10

NEW VERSION OVERVIEW - Take Command 10.0

This is a summary of the compatibility fixes and new features. For complete details, see the appropriate topics in this help file.

The new features that are supported in TCC/LE (including the TCC/LE component of Take Command/LE) are marked with a *.

Feature List:

- * Take Command and TCC startup is faster.
- * Most operations in Take Command and TCC are faster.

The quick help displayed on the TCMD status bar will now also identify aliases and display their value.

Holding down the Ctrl key while scrolling the mouse wheel will now change the font size in the Take Command tab windows. Note that not all apps will be happy about you randomly changing their font (and thus the console window size)!

Increased the maximum number of TCMD startup tabs from 10 to 20.

You can now drag files from the desktop to the List View window.

Added an option to set the popup window font for Take Command and TCC (Windows page of their configuration dialogs).

Added support for specifying attributes (/A:...) and ranges in the TCMD Filter combobox. (The syntax is the same as in TCC.) The attribute and range switches must come before the filename wildcards (or regular expressions).

The TCMD filter combobox now saves its entries (up to 10) and restores them when TCMD restarts.

When starting a tab, if the filename is "TCC.EXE" or "TCC" and there is no path specified and TCMD cannot find the filename in the path, it will default to running TCC.EXE in the TCMD installation directory.

The "Up" button on the Take Command toolbar will now take you to "Computer" if you're already at the root directory of a drive.

The Take Command "Find Files" dialog "Edit" button will now try to open the file with its associated app. If that fails, Take Command will use Notepad.

Added support for array variables. See SETARRAY / UNSETARRAY, SET, @ARRAYINFO,

@EXECARRAY, and @FILEARRAY for details.

Added internal support for Python (.py). You must enable Python in the OPTION / Startup dialog.

Added an "in-process" pipe. This works like the old DOS pipes, by creating a temporary output file, redirecting STDOUT to that file, and then redirecting the temp file to STDIN of the following command.

The syntax is:

```
command1 !| command2
```

(This is the same as doing "command1 > temp.dat & command2 < temp.dat", but is easier to type & to read.) There are some disadvantages to using this type of "pseudo-pipe" -- it will usually be slower than a true pipe; it will use some disk space for its temp file; and "command2" will not be started until "command1" has exited.

Added "here-string" input redirection to send a string to the program's standard input. The syntax is:

```
cmd <<< this is some input text
```

The popup windows (history, directory history, fuzzy directory search) now save their new size & position when moved.

- * File & disk size variable functions (@DISKFREE, @FILESIZE, etc.) now support terabytes (trailing t or T) and petabytes (trailing p or P).
- * Size ranges now support terabytes (trailing t or T) and petabytes (trailing p or P).

Date ranges now support ages for the first and/or second parameter.

- * The Take Command and TCC configuration dialogs now display the name of the active TCMD.INI file in the titlebar. (Hopefully this will reduce some of the "bug" reports when people aren't writing to the TCMD.INI they think they are!)

If the cursor is hidden in a console running in tab window, the tab window cursor will now also be hidden.

Added a combo box to the Take Command and TCC configuration dialogs to allow you to select a language dll (default, English, French, or German).

HTTP and HTTPS addresses in TCMD and TCC will now have embedded spaces converted to "%20" before sending it to the server.

The TCCTABHERE.BTM file will create a "TCC tab window here" prompt in the Folders and List View context menus.

You can now mark text in a TCC tab window using the TCC alt-cursor keystrokes and copy/paste the text using the TCMD edit menu and right-click context menu. (Though it's easier & a lot more powerful to use the command window instead!)

- * Fuzzy directory searching is now 30-50% faster.

A trailing & on the command line (with preceding white space) will start the command line in a

detached process (like Linux). (This is the same as prefixing "DETACH" to the command, but a little easier to type and more natural for Linux users.)

Dropped support for obsolete & unsupported REXX interpreters -- Take Command now only supports ooREXX (Object REXX).

Added Linux-style numeric variable expressions - %((...)) will evaluate and substitute the expression. For example:

echo %((3+5)) is the answer.

(For non-Linux users, this is the same as @EVAL[3+5] but is slightly easier to type.)

Startup Options:

.INI Directives:

BackgroundImage=filename - BMP file to use as TCMD tab window background.

BatchAliases=YES|no - if set to NO, TCC won't try to expand command aliases when in a batch file. (Directory aliases will still be expanded.)

ClosselfNoTabs=YES|no - if set to NO, TCMD won't close if there are no tab windows open. (See the Take Command "Advanced" tab.)

- * CompleteHidden - has been replaced by CompleteHiddenFiles and CompleteHiddenDirs. CompleteHidden will still be recognized if set in TCMD.INI (CompleteHidden=Yes will now be converted to a CompleteHiddenFiles=Yes and a CompleteHiddenDirs=Yes.) It has been removed from the OPTION dialog.
- * CompleteHiddenFiles=yes|NO - if set to YES, tab completion will look for hidden files and system files as well as normal files.
- * CompleteHiddenDirs=yes|NO - if set to YES, tab completion will look for hidden directories as well as normal directories.

Python=yes|NO - enable internal Python support.

Command Line Editing:

Ctrl-F (expand aliases) will now also expand any directory aliases on the line. (Useful when passing directory aliases to external apps.)

New Commands:

DISKMONITOR - monitor the free disk space. If it drops below the specified size, DISKMONITOR will execute the specified command.

DISKMONITOR [/C] drive size command

For example:

DISKMONITOR C: 2Gb sendmail bob@bob.com "Disk Status" Drive C: is full!

The drive can also be a sharename. The size format is the same as that used for size ranges (i.e., either a number or a number with an appended k, K, m, M, g, G, t, or T).

IDE - start the Take Command IDE / debugger with the specified files loaded into tab windows.

SETARRAY - define array variables. You can define up to 4-dimensional arrays. The syntax is:

SETARRAY name[a[,b[,c[,d]]]] [...]

where a, b, c, and d are the sizes. For example, to define a 5-row by 10-column array:

setarray array1[5,10]

(The array elements are addressed in base 0, so to reference this array you would use 0-4 for the rows and 0-9 for the columns.)

To set the variable elements, use the SET command (see below).

If you don't enter any arguments, SETARRAY will display the currently defined arrays. If you don't enter any dimensions, SETARRAY will display the definition for that array. You can use wildcards in the array name.

SETERROR - set the %ERRORLEVEL value and the last-error code in Windows to the specified value. The syntax is:

SETERROR n

STATUSBAR - write text to the Take Command status bar. The syntax is:

STATUSBAR text

TASKBAR - call the Windows Taskbar to to display dialogs or to manipulate the top level windows.
The syntax is:

TASKBAR command

Where "command" is one of the following:

Cascade	Cascade all top level windows
Computers	Display the Find Computers dialog (requires Active Directory Domain Services)
Control	Display the Control panel
Customize	Display the Customize Taskbar dialog
Date	Display the Date and Time dialog
Desktop	Show the Windows desktop
Help	Display the Help and Support Center dialog
HTile	Horizontally tile all top level windows
Lock	Toggle the taskbar lock

Min	Minimize all windows
Max	Maximize all windows
Printers	Display the Printers and Faxes dialog
Properties	Display the Taskbar Properties dialog
Run	Display the Run dialog
Search	Display the Search dialog
Shutdown	Display the Shut Down Computer dialog
Start	Display the Start Menu
Task	Display the Windows Task Manager dialog
VTile	Vertically tile all top level windows

UNSETARRAY - remove array variables. The syntax is:

UNSETARRAY [/Q] name [...]

/Q - quiet (don't display an error for a non-existent array)

You can use wildcards in "name".

Commands:

ACTIVATE - added a new option:

/FLASH=type,count - flash the specified window. The arguments are:

type - type of flash; one or more of the following values:

- 0 - stop flashing
- 1 - flash the window caption
- 2 - flash the taskbar button
- 4 - flash continuously until WINDOW is called again with the /FLASH type set to 0
- 12 - flash continuously until the window comes to the foreground (cannot be used with 4)

count - the number of times to flash the window

ATTRIB - added new options:

/N - do not actually change the attributes

/NE - no error messages

/NJ - no junctions (only useful with /S)

/S+n - start 'n' levels down from the source directory.

- * CD / CDD - fuzzy directory searching is substantially faster (up to 50%).

CD / CDD - now allow a forward slash at the beginning of a directory name > 1 character (for unreconstructed Linux users). (In CDD, this may conflict with the multicharacter options; in that case, the options will win out over the directory.)

COPY - if you specify the /C or /U options, COPY will no longer return an error result (2) in %? if no files match.

COPY - added new options:

/CF - copy the source to the target if the target file exists and is more than 2 seconds older than the source.

/UF - copy the source to the target if the target file doesn't exist or is more than 2 seconds older than the source.

/S+n - start 'n' levels down from the source directory.

/Sx - subdirectory copy to a single target directory (implies /S). For example, to copy all of the .EXE files in "c:\files" and all of its subdirectories to the directory "d:\exefiles":

```
copy /sx c:\files\*.exe d:\exefiles\
```

DATE - added new options:

/Fn - where *n* is the format to use:

```
0 : "Mon Jan 1, 2009"  
1 : " 1/01/09"  
2 : "Mon 1/01/2009"
```

/U - display or set the UTC date

DEL - added a new option:

/S+n - start 'n' levels down from the source directory.

- * DEL - reformatted the summary to support up to 100 Tb partition sizes.

DELAY - added a new option:

/F - flush the keyboard buffer after the delay ends

DIR - added a new option:

/S+n - start 'n' levels down from the specified directory.

- * DIR - reformatted the summary to support up to 100 Tb partition sizes.
- * DIRHISTORY /R - is now 500% faster.

DIRHISTORY /Tn - display the last n lines of the directory history. If n is negative, skip the first -n lines of the directory history.

DO - added new options:

DO var in /P command ... - parse the output of a command:

/S+n - start 'n' levels down from the source directory.

DO - the LEAVE option now accepts an optional parameter to leave nested DO's:

LEAVE n -

ESET - removed the /W option (a result of the batch debugger changes).

FFIND /W - the "Edit" button will now try to open the file with its associated app. If that fails, FFIND will use Notepad.

FFIND - added a new option:

/S+n - start 'n' levels down from the source directory.

FOLDERMONITOR - added a new option:

/U - don't set the trigger until the file is unlocked.

- * FREE - reformatted the output to support up to 100 Tb partition sizes.

GLOBAL - added a new option:

/S+n - start 'n' levels down from the current directory.

IFTP - a /C will now prevent an automatic reconnection if you try something like a "dir ftp:" after the IFTP /C.

- * HISTORY /R - is now 500% faster.

HISTORY -- added new options:

/Tn - display the last n lines of the history. If n is negative, skip the first -n lines of the history.

/V - display the history in reverse order.

INKEY - added support for array variables.

INPUT - added a default value for the /E option. The syntax is:

/E"value"

If the environment variable doesn't exist, INPUT will display the default value for editing.

INPUT - added support for array variables.

KEYBD - added a new option:

/Kn - disable (0) or enable (1) the keyboard. (You can also reenable a disabled keyboard with Ctrl-Alt-End.)

LIST - The "E(edit)" option will now first try to edit the file using the editor associated with that filetype (if any). If that fails, LIST falls back to its previous behavior (using the Editor .INI directive if it exists or Notepad.exe if it doesn't).

LIST - added a new option:

/F - display the contents of the console screen buffer.

MOVE - added new options:

/CF - move the source to the target if the target file exists and is more than 2 seconds older than the source.

/UF - move the source to the target if the target file is more than 2 seconds older than the source.

/S+n - start 'n' levels down from the source directory.

/Sx - subdirectory move to a single target directory (implies /S). For example, to move all of the .EXE files in "c:\files" and all of its subdirectories to the directory "d:\exefiles":

```
move /sx c:\files\*.exe d:\exefiles\
```

MSGBOX - added new options:

/L - limit the maximum message box width to no more than 1/3 the screen width

/V - display the message box in the Vista style (the message background will be the current window color, the buttons will be right-justified and slightly bigger, and the position of icon and message will be adjusted.)

MSGBOX now supports Ctrl-C to copy the contents of the message box to the clipboard.

- * OPTION - removed the popup windows from the Windows tab, as they are now auto-saved whenever the windows are moved or resized.

ON - added new options:

ON CONDITION test command ... - execute the command if the test is true. The test can be any valid test that is valid in IF.

ON RESUME command - execute the command when the system resumes from a suspension (sleep or hibernation).

ON SUSPEND command - execute the command when the system is going to sleep or hibernation.

- * PDIR - added support for quoting filenames (if necessary). The syntax is:

`pdir [/f[pnq]]` - 'q' will cause the name and/or path to be double quoted if it contains white space or special characters.

- * PDIR - will now suppress empty lines (for example, if you have an @IF conditional in PDIR and not every matching file results in output).

PDIR - added a new option:

/S+n - start 'n' levels down from the specified directory.

PROCESSMONITOR - added the HUNG test.

- * REN - added some additional checking to REN /N to see if the rename would actually succeed (i.e., checking for things like the target filename already existing).

SET - added support for setting array variables. For example, to define a 5-row by 10-column array, you would first use SETARRAY (see above):

```
setarray array1[5,10]
```

To set the array values (0-based), the syntax is:

```
set array1[a[,b[,c[,d]]]]
```

For example:

```
set array1[0,0]=Bob  
set array1[0,1]=Bob's Job
```

To expand the array variable:

```
echo Name is %array1[0,0] and job is %array1[0,1]
```

SETDOS - the /Y option (which has been deprecated since 7.0) has been removed.

- * START - changed the /Affinity option to match the new CMD.EXE behavior (in Vista and XP64). It now takes a hex argument for the processor mask -- i.e., to set the affinity for cpu's 1 and 4, set /affinity=5.

SYNC - added new options:

/S+n - start 'n' levels down from the specified directory.

/Y - suppress the prompt if you have the "COPY Prompt on Overwrite" option set.

/Z - overwrite read-only files

TCTOOLBAR - added new options:

/I - reset toolbar to definition in TCMD.INI.

/W filename - save the toolbar to the specified file.

TIME - added a new option:

/U - display or set the UTC time

TOUCH - added a new option:

/S+n - start 'n' levels down from the specified directory.

TYPE - added new options:

/X - display the file in hex

/XS - display the file in hex, using spaces instead of periods for non-printable characters.

WINDOW - added a new option:

/FLASH=type,count - flash the TCC or TCMD window. The arguments are:

type - type of flash; one or more of the following values:

0 - stop flashing

1 - flash the window caption

2 - flash the taskbar button

4 - flash continuously until WINDOW is called again with the /FLASH type set to 0

12 - flash continuously until the window comes to the foreground (cannot be used with 4)

count - the number of times to flash the window

Internal Variables:

_elevated - (Vista and above) - returns 1 if the TCC process is elevated

_ide - returns 1 if in the IDE / debugger

_lastdir - previous directory (from directory history)

_selected - selected text in the current tab window. (This is normally only useful in toolbar buttons or key aliases, as the selected text will revert to normal on a keystroke.)

_tctabs - current number of Take Command tab windows (0 if not in TCMD).

_vermajor - TCC major version

_verminor - TCC minor version

_version - TCC version in "major.minor" format (i.e., "10.0").

_xwindow - width of the Take Command or TCC window in pixels

_ywindow - height of the Take Command or TCC window in pixels

Variable Functions:

@ARRAYINFO[arrayname,option] - returns information about the specified array.

arrayname - name of the array (defined by SETARRAY) to query

option - the type of information:

0 - total number of dimensions

1 - # of elements in the first dimension

2 - # of elements in the second dimension

- 3 - # of elements in the third dimension
- 4 - # of elements in the fourth dimension
- 5 - total number of elements

@BALLOC[size] - alloc a buffer for binary operations. The function returns a handle to the buffer (which must be used for the subsequent binary functions). The only limit on the number & size of the binary buffers is the amount of virtual memory available.

@BFREE[handle] - free a binary buffer (previously allocated by @BALLOC).

@BPEEK[handle,offset,size] - read a value from a binary buffer.

handle - a binary handle from @BALLOC

offset - the byte offset in the buffer

size - the size of the value (in bytes):

- 1 - character
- 2 - short
- 4 - int
- 8 - int64

@BPEEK returns the value read

@BPEEKSTR[handle,offset,type,length] - read a string from a binary buffer.

handle - a binary handle from @BALLOC

offset - the byte offset in the buffer

type - the string type:

- a - ASCII
- u - Unicode

length - the maximum number of characters to read

@BPEEKSTR returns the string

@BPOKE[handle,offset,size,value] - write a value to a binary buffer.

handle - a binary handle from @BALLOC

offset - the byte offset in the buffer

size - the size of the value (in bytes):

- 1 - character
- 2 - short
- 4 - int
- 8 - int64

value - the value to poke

@BPOKE returns 0 on success.

@BPOKESTR[handle,offset,type,string] - write a string to a binary buffer.

handle - a binary handle from **@ALLOC**

offset - the byte offset in the buffer

type - the type of the string to write:

a - ASCII

u - Unicode

string - the string to poke

@BPOKESTR returns 0 on success.

@BREAD[handle,offset,filehandle,fileoffset,length] - read from a file to a binary buffer.

handle - a binary handle from **@ALLOC**

offset - the byte offset in the buffer

filehandle - a file handle opened for reading (from **@FILEOPEN**)

fileoffset - the read offset (from the current file position)

length - number of bytes to read

@BREAD returns the number of bytes actually read.

@BWRITE[handle,offset,filehandle,fileoffset,length] - write from a binary buffer to a file

handle - a binary handle from **@ALLOC**

offset - the byte offset in the buffer

filehandle - a file handle opened for writing (from **@FILEOPEN**)

fileoffset - the write offset (from the current file position)

length - the number of bytes to write

@BWRITE returns the number of bytes written

@EVAL - added support for array names w/o a leading %.

@EVAL - added new operators and functions:

!a - return the inverse not (i.e., !0 = 1, !5 = 0)

a>b - return 1 if a is greater than b

a<b - return 1 if a is less than b

fact(a) - return the factorial

ceil(a) - return the ceiling

floor(a) - return the floor

`abs(a)` - return the absolute value
`gcd(a b)` - return the greatest common divisor
`lcm(a b)` - return the least common multiple

`ror(value shift precision)` - rotate right
`rol(value shift precision)` - rotate left
 value - integer value to rotate
 shift - the number of bits to shift
 precision - the size of "value" in bits

For example, to rotate the 32-bit integer "123" 2 bits to the right:

```
%@eval[ror(123 2 32)]
```

`@EXECARRAY[array,command]` - execute the specified command and store the resulting lines in the specified array. (You must define the array before running `@EXECARRAY`.) For example:

```
setarray aresult[10]
echo %@execarray[aresult,dir /u] >& nul
```

`@EXECARRAY` will read the number of lines specified in the array size definition.

`@EXECSTR` - added new option for the line to return. The syntax is:

```
@EXECSTR[[n.]command]
```

where "n" is the line you want (base 0). For example, to return the third line returned by VER /R:

```
echo %@execstr[2,command]
```

`@FILEARRAY[array,filename]` - read a file and store the lines in the array. (You must define the array before running `@FILEARRAY`.) For example:

```
setarray aresult[10]
echo %@filearray[aresult,test.dat]
```

`@FILEARRAY` will return the number of lines read. `@FILEARRAY` will not read more than the number of lines specified in the array size definition.

`@FILEHANDLE[handle]` - returns the filename for the specified file handle (opened with `@FILEOPEN`).

`@FILES` - added /S+n option to start 'n' levels down from the specified directory.

`@FILESIZE` - added /S+n option to start 'n' levels down from the specified directory.

`@FILTER[chars,string]` - removes any characters in "string" that aren't in "chars". For example, to remove all non-numeric characters from a variable:

```
%@filter[0123456789,%var]
```

`@FOLDERS[directory]` - returns number of matching folders.

@ISFLOAT[string] - returns 1 if the string is composed only of numeric characters, a decimal separator, and an optional sign and/or thousands separator(s).

@ISLOWER[string] - returns 1 if the string is composed only of lower case letters.

@ISUPPER[string] - returns 1 if the string is composed only of upper case letters.

* **@LINE** - is now 700% faster.

* **@LINES** - is now 700% faster.

@MX[address] - return the email server for the specified user address.

@PID[filename] - returns the PID for specified name (or 0 if no match). If you have multiple copies of the same executable running, @PID will return the first one it finds.

@PYTHON[command] - execute the Python string. The Python interpreter is persistent; if you want to reset it pass an empty string to @PYTHON.

@REGCOPYKEY[source,target] - copy a registry key.

@REGCREATE, **@REGDELKEY**, **@REGEXIST**, **@REGQUERY**, **@REGSET**, **@REGSETENV**, and **@REGTYPE** - added an option to access the 64-bit registry in Win64. If you append "_64" to the HKEY name, TCC will access the 64-bit registry instead of the 32-bit registry. For example:

@regcreate["HKLM_64\Software\Company\Product\User"]

@SERVICE[service,info] : Returns information about the specified service.

service - the service name to query

info - the information you want:

- 1 The type of service
- 2 The current state of the service
- 3 The control codes the service accepts and processes in its handler function.
- 4 Returns the check-point value the service increments to report its progress during a lengthy start, stop, pause, or continue operation.
- 5 Returns the estimated time required for a pending start, stop, pause, or continue operation (in milliseconds).

@SNAPSHOT - added support for multiple monitors when using the DESKTOP argument.

@WINCLIENTSIZE[title] - returns the client window size in the format:

height,width

@WINPID[title] - returns the process ID for the window.

@WINSIZE[title] - returns the window size in the format:

height,width

Batch Debugger:

The IDE / batch debugger is all new, and includes multiple tabbed editing windows and tabbed / dockable watch & variable windows.

The IDE now supports themes (Options / Theme).

In addition to batch files (.BTM, .BAT, and .CMD) the syntax coloring in the editor also supports editing C++, INI, Javascript, LUA, Pascal, Python, SQL, VBScript, and XML files.

The IDE maintains a list of the recently edited files.

There are new options in the File menu (Save All, Close All, Print Preview).

There are a number of new options in the Edit / Advanced menu (tabify / untabify, make selection uppercase / lowercase, collapse / expand, and view whitespace).

The "Pause" button / menu entry pauses debugging at the completion of the current command line.

The watch window now also supports internal variables, variable functions, and user-defined functions.

The IDE status bar is customizable (right click).

3 Take Command

This section provides a general description of **Take Command** operation.

- [Starting Take Command](#)
- [The Take Command Interface](#)
- [Configuration Options](#)
- [The Take Command Interface](#)
- [Take Command and TCC Integration](#)

3.1 Starting Take Command

You will typically start **Take Command** from a Windows shortcut, located:

- on the desktop, or
- in the **Programs** section of the **Start** menu (including its **Startup** subdirectory).

You may also start it from the **Start / Run** dialog.

The installation software will optionally create both a **Take Command** folder or group (in the **Programs** section of the **Start** menu) and a desktop object (shortcut) which starts **Take Command**. Usually these are sufficient, but if you prefer, you can create multiple desktop objects or items to start **Take Command** with different startup commands or options, or to run different applications in the tab windows.

Each item or icon represents a different **Take Command** window. You can set any necessary command line parameters for **Take Command** such as a program to run in a tab window, and the name and path for the [.INI file](#). See [Take Command Startup Options](#) for more information on startup command line options.

When you configure a **Take Command** item, place the full path and name for the file in the Command Line field, and put any startup options that you want passed to **Take Command**. For example:

Command Line:	C:\Program Files\JPSoft\TCMD\TCMD.EXE
Working directory:	C:\

You do not need to use the Change Icon button, because **TCMD.EXE** already contains icons.

Each Windows program has a command line which can be used to pass information to the program when it starts. The command line is entered in the Command Line field for each shortcut or each item in a Program Manager group (or each item defined under another Windows shell), and consists of the name of the program to execute, followed by any startup options.

The **Take Command** startup command line does not need to contain any information. When invoked with an empty command line, **Take Command** will configure itself from the [TCMD.INI file](#), and then display a prompt and wait for you to type a command. However, you may add information to the [startup command line](#) that will affect the way **Take Command** operates.

3.1.1 Take Command Startup Options

The **Take Command** command line includes the program name with drive and path, followed by any options. For example:

```
"c:\program files\jpsoft\tcmd13\tcmd.exe" @c:\tcmd\tcmd.ini
```

There are several **Take Command** startup options. The complete syntax for the **Take Command** startup command line is (all on one line):

```
d:\path\tcmd.exe [[/]@d:\path\infile] [//directive=value...] [/D d:  
 \path] [/N] [/C command] [/T [d:\path\]program]
```

(Do not include the square brackets shown in the command line above. They are there to indicate that the items within the brackets are optional.)

The command line must start with the full **Take Command** path and executable name (**TCMD.EXE**):

```
d:\path\tcmd.exe
```

The additional items below may be included on the command line:

```
@d:\path\infile OR  
/@d:\path\infile
```

This option sets the path and name of the [.INI file](#). You don't need this option if

- 1) your [.INI file](#) is named **TCMD.INI**, and
- 2) it is in one of the following directories:
 - 2.1) the same directory as **Take Command**

2.2) the %localappdata% directory

This option is most useful if you want to start the program with a specific and unique [.INI file](#).

To start **Take Command** without any [.INI file](#), you can create an empty file and specify it as your [.INI file](#).

To get around a Windows limitation that causes the displayed command line of a shortcut to be truncated when a parameter begins with @, you can use the alternative syntax

```
/@d:\path\inifile
```

Take Command will skip the leading forward slash.

Options:

//directive=value

This option tells **Take Command** to treat the text appearing between the // and the next space or tab as an initialization directive. The directive should be in the same format as a line in [TCMD.INI](#), but may not contain spaces, tabs, or comments. This option may be repeated. It is a convenient way to place a few simple directives on the startup line without having to modify or create a new [.INI file](#).

- /C** Run the specified command in a new **TCC** tab window. If there is already a **Take Command** session running, /C creates a new tab in the existing **Take Command** rather than starting a new session. /C must be the last option on the command line (otherwise **Take Command** can't tell if additional options belong to **Take Command** or the command to run in the **TCC** tab).
- /D** Start **Folders** and **List View** in the specified directory.
- /N** Don't load **TCMD.INI** (useful when trying to isolate configuration problems).
- /NT** Don't load the default startup tabs (usually only useful when combined with /C or /T).
- /T** You can specify the program to start in the first tab with the /T option:

```
d:\path\tcmd.exe /t d:\path\program
```

If there is already a **Take Command** session running, /T creates a new tab in the existing **Take Command** rather than starting a new session.

/T must be the last option on the command line (otherwise **Take Command** can't tell if additional options belong to **Take Command** or the program to start in the tab).

If you have [Startup Tabs](#) defined, **Take Command** will display them following the tab created by /T.

3.2 Take Command Interface

The **Take Command** Window

- [The Take Command Window](#)
- [Menus](#)
- [Tool Bars](#)
- [Folders](#)
- [List View](#)
- [Command Input](#)
- [Tab Windows](#)
- [Status Bar](#)
- [Keyboard Shortcuts](#)
- [Context Menus](#)
- [Using the Scrollback Buffer](#)
- [Highlighting and Copying Text](#)
- [Take Command Dialogs](#)

Starting Applications

- [Starting Windows Applications](#)

Take Command and the Windows Environment

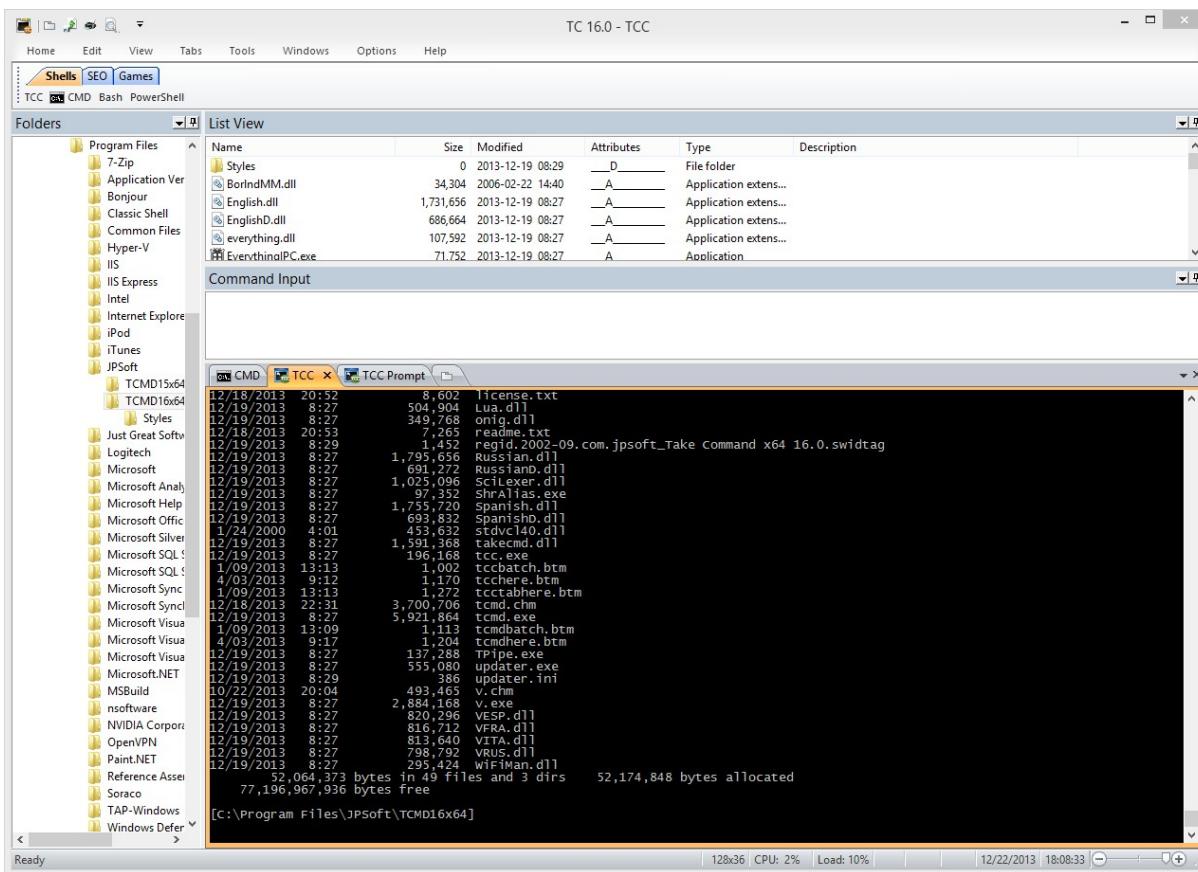
- [Resizing the Take Command Window](#)
- [Drag and Drop](#)

Take Command and TCC

- [Take Command and TCC Integration](#)

3.2.1 Take Command Window

The **Take Command** window has eight parts:



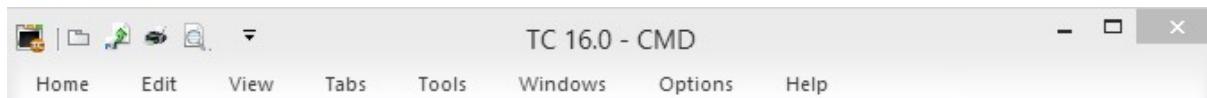
1. The **Title Bar** is the same as the one used in most Windows applications, with a control menu button on the left and the minimize, maximize, and close buttons on the right. You can also adjust the size of the **Take Command** window using standard window techniques, but see [Resizing the Take Command Window](#) for information about how **Take Command**'s display changes when you do so.
2. See [Take Command Menus](#) for details about the **Menu Bar** and all of its submenus.
3. The Tabbed Toolbar is an optional tool bar that you can use to execute internal or external commands, aliases, or batch files with the click of a mouse (or an accelerator key). You can define up to 20 toolbar tabs, each with up to 50 buttons. To create buttons for the tab toolbar, right click on the toolbar and select "Add Button" or "Add Tab". This selection displays the [tool bar dialog](#). You can also configure the tab toolbar from **TCC** with the [TCTOOLBAR](#) command.
4. The **Folders** window shows a tree view of your desktop.
5. The **List View** window shows the contents of the item selected in the **Folders** window. You can display the **List View** window in a number of ways (Large Icons, Small Icons, List, and Details views).
6. The **Command Input** window allows you to create and edit commands before sending them to the active tab window. This is not limited to entering something at the command prompt; you

can feed strings anywhere a console app is expecting input. You can scroll back to previous lines, edit, and reexecute them. The **Command Input** window also has full undo/redo (up to 31 levels), drag and drop, and both mouse and keyboard text selection.

7. The **Tab Windows** run the **Take Command Console**, or any other Windows console application (including CMD, PowerShell, or bash). You can use the scroll bars or the **Alt** cursor keys to view text that has scrolled through the window. You can also save the contents of a tab window and scrollback buffer to a file, copy text from a tab Window to the clipboard, and copy text from the clipboard or from the tab window scrollback buffer to the command line. See [Highlighting and Copying Text](#) for information about saving and retrieving text in the tab window and [The Command Line](#) for complete details about using the **Take Command** console command line.
8. Finally, the **Status Bar** at the bottom of the **Take Command** window displays information about your system:
 - ▶ Tooltips for the menu selections
 - ▶ The tab window size (columns x rows)
 - ▶ The CPU usage (0 - 100%)
 - ▶ The memory load (0-100%)
 - ▶ The state of the Caps Lock key
 - ▶ The state of the Num Lock key
 - ▶ The state of the Scroll Lock key
 - ▶ The current date
 - ▶ The current time
 - ▶ A slider control to change the **Take Command** transparency

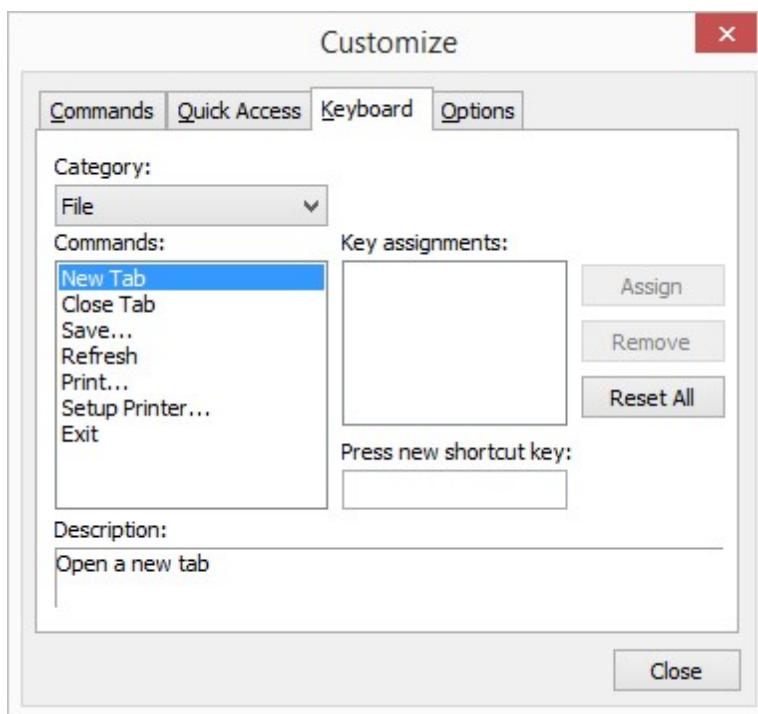
If you find the "I-Beam" cursor in the **Take Command** tab windows difficult to see, you can change it in the [Tabs](#) page of the configuration dialog to force the use of an arrow cursor in all parts of the window.

3.2.1.1 Menus



Like many Windows applications, **Take Command** displays a ribbon menu along the top of the **Take Command** window. The default behavior of the ribbon menu is to drop down when you select a menu entry, or you can specify that it always be displayed. (This requires additional screen space.) To select a particular menu item, click once on the menu heading, or use **Alt-x** where **x** is the underlined letter on the menu bar (for example, **Alt-F** displays the **File** menu). You can also select a menu by pressing **Alt** or **F10** and then moving the highlight with the cursor keys.

The **Take Command** Quick Options toolbar and ribbon menu bar are customizable. To customize Quick Options or the ribbon menu (including accelerator keys), click on the button on the right side of the Quick Options toolbar.



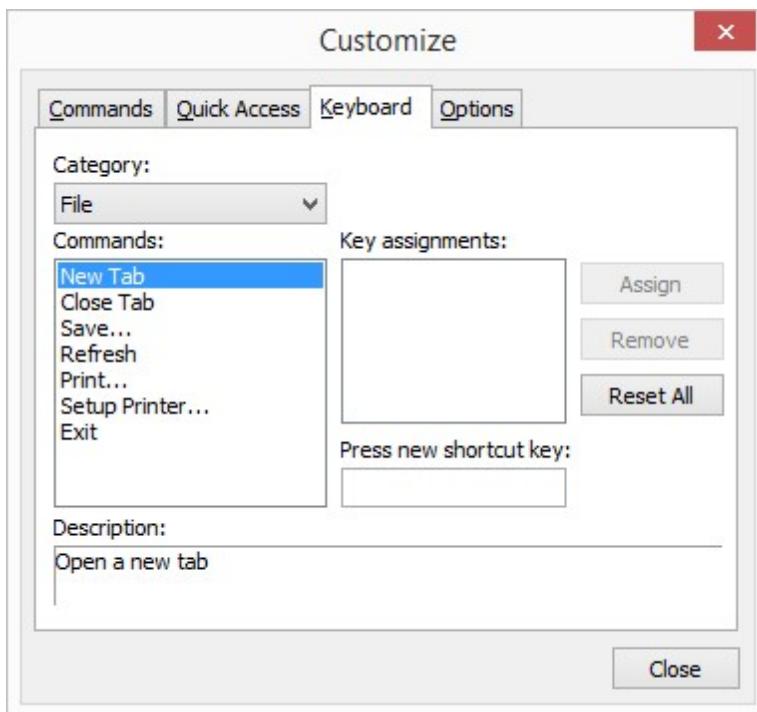
The menu bar entries allow you to select a variety of **Take Command** features:

[Home](#)
[Edit](#)
[View](#)
[Tabs](#)
[Windows](#)
[Options](#)
[Help](#)

3.2.1.1.1 Quick Options

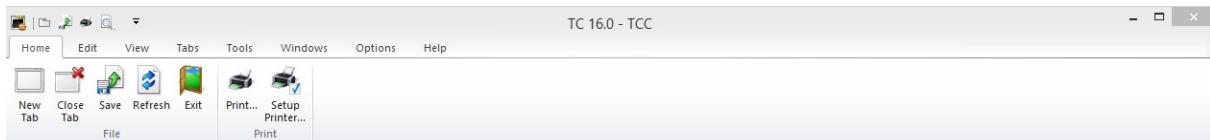
The Quick Options toolbar is on the left side of the caption (title) bar. The Quick Options toolbar enables / disables the Tabbed Toolbar, and minimizes or displays the **Take Command** menu ribbon.

Quick Options can also display the Customize dialog, which allows you to add commands to Quick Options, drag & drop icons to the tabbed toolbar, and change the keyboard shortcuts.



3.2.1.1.2 Home

The Home menu allows you to create new tabs, save or print the screen buffer, or exit **Take Command**.



New Tab

Opens the default **Take Command Console** command processor in a new tab window. (If you want to run a different application in a new tab, use the Run menu entry in the [Tabs](#) menu.)

Close Tab

Closes the active tab window.

Save to File...

Saves the contents of the current tab window's scrollback buffer to a file. A Save As dialog box appears in which you can enter the name of the file that you wish to use.

Refresh

Redraws everything in the current **Take Command** window (use this selection if the display appears incorrect).

Exit

Ends the current **Take Command** session.

Print...

Sends the contents of the current tab window's scrollback buffer to the printer. A Print dialog box appears in which you can choose the portion of the screen buffer you wish to print.

Printer Setup...

Displays a standard printer setup dialog box. The options available in the dialog box depend on the printer driver(s) you are using.

3.2.1.1.3 Edit

The Edit menu allows you to copy text between the **Take Command** windows and the Windows clipboard. You can also access the clipboard in **TCC** with [redirection](#) to or from the CLIP: device, or with the [@CLIP](#) variable function.



To use the Cut, Copy, or Delete commands, you must first select a block of text with the mouse, the keyboard, or with the Select All command, below. If you hold down the right mouse button while you select a block of text, that block will be copied to the clipboard automatically when you release the button.

For more information on copying text see [Highlighting and Copying Text](#).

Copy

Copies selected text from the command line or scrollback buffer to the clipboard.

Paste

Copies text from the clipboard to the command line. If the text you insert contains a line feed or carriage return, the command line will be executed just as if you had pressed Enter. If you insert multiple lines, each line will be treated like a command typed at the prompt.

Copy + Paste

Copies the selected text from the scrollback buffer directly to the command line.

Copy + Paste + Run

Copies the selected text from the scrollback buffer directly to the command line and executes the resulting command line.

Paste + Run

Copies text from the clipboard to the command line and executes the resulting command line.

Select All

Marks the entire contents of the scrollback buffer as selected text.

Undo

Undo the last action in the Command Input window.

Redo

Redo the last action in the Command Input window.

Insert Filename

Displays the Windows file selection dialog and puts the selected filename at the current position on the command line.

Insert Folder

Displays the Windows folder selection dialog and puts the selected directory name at the current position on the command line.

Console Mouse

Send mouse moves and clicks to the console window. This option is specific to each tab, and is rarely necessary because few console apps use the mouse. You can also toggle this option with **Ctrl-M** in the tab window.

Find

Search the scrollback buffer of the current tab window using [regular expressions](#).

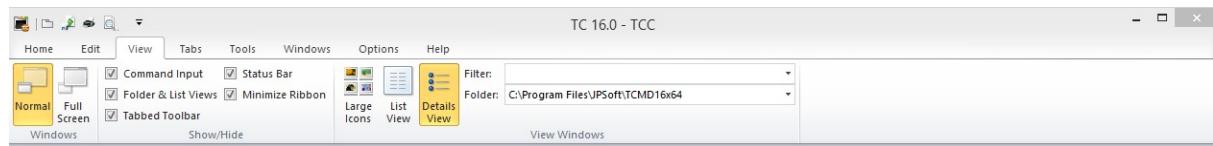
Replace

Search and replace text in the Command Input window.

Find Files

Search for files or text within files.

3.2.1.1.4 View



Normal

Restores the Take Command window to its normal size, and restores the ribbon, Folder, List View, and Command Input windows to their default state and size.

Full Screen

Maximizes the tab window size by autohiding the ribbon, Folders, List View, and Command Input windows, and maximizing the Take Command window.

Command Input

Show or hide the [Command Input](#) window.

Folder and List Views

Show or hide the [Folders](#) and [List View](#) windows. If you want to have more room for your tab windows, you can set the **Folders** and **List View** windows to AutoHide (i.e., they will be collapsed to a single tab label when not in use). If you don't want the view windows to appear at all, you can toggle them off with this option.

Tabbed Toolbar

Show or hide the [tabbed toolbar](#).

Status Bar

Show or hide the [status bar](#).

Minimize Ribbon

Show or autohide the Take Command ribbon menu.

Large Icons

Display items in the List View using large icons.

List

Display items in the List View as a list.

Details

Display directory entries in the List View with a DIR-style display (Name, Size, Date/Time, Attributes, Type, and Description).

Filter

You can enter an expression to use to filter the directory entries displayed in the List View. The **Filter** combo box allows you to enter [wildcards](#) or [regular expressions](#) to filter the directories and files displayed in the **List View** window. You can enter multiple wildcards by enclosing each argument in double quotes and separating them with a comma. For example:

`"*.cmd", "*.txt", "*.doc"`

To enter a regular expression in the **Filter** combo box, precede it with two colons. For example:

`::/c`

You can specify [attributes](#) (/A:...) and [ranges](#) in the **Filter** combo box. (The syntax is the same as in **TCC**.) The attribute and range switches must come before the filename wildcards or regular expressions.

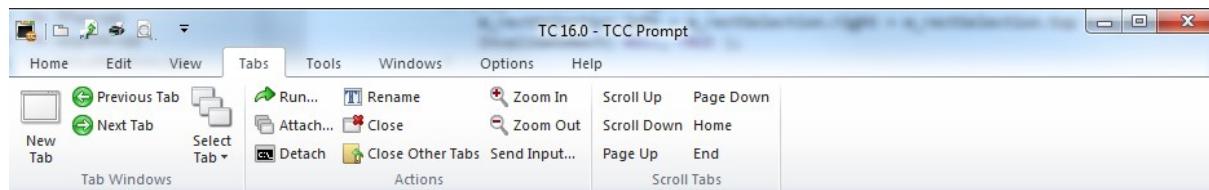
You can also set the filter from **TCC** with the [TCFILTER](#) command.

The **Filter** combo box will save its last 10 entries in each session and restore them when **Take Command** restarts.

Folder

The **Folder** combo box changes the current directory for the **Folders** and **List View** windows.

3.2.1.5 Tabs



New Tab

Opens the default command processor in a new tab window. **Take Command** defaults to the program specified in your COMSPEC environment variable; if that isn't found then **Take Command** will start **TCC.EXE**.

Previous Tab

Switch to the next tab window.

Next Tab

Switch to the previous tab window.

Select Tab

Select a tab window to activate from a drop-down list.

Run

Displays the [run dialog box](#) from which you can run an application or batch file. **Take Command** remembers the commands you have run from this dialog in the current session. To select from this list click on the drop-down arrow to the right of the "Command Line" field, or press the down-arrow. If the filename is *TCC.EXE* or *TCC*, there is no path specified, and **Take Command** cannot find the filename in the path, it will default to running *TCC.EXE* in the **Take Command** installation directory.

Attach

Displays a popup dialog of all of the console sessions that are **not** already displayed in a **Take Command** tab window, and allows you to select one or more to convert to a **Take Command** tab. You can select multiple sessions by clicking on individual entries and then on **OK**, or you can select a single session by double clicking on the entry.

Detach

Disconnects the current tab window from **Take Command** and displays it on the desktop.

Rename

Rename the current tab text (and the console title). If you set the title with this option, **Take Command** will not change the tab title if the application subsequently changes the console title. If you set the title to an empty string, the title will revert to that set by the console application.)

Close

Closes the current tab window.

Zoom In

Increase the tab window font size by one point.

Zoom Out

Decrease the tab window font size by one point.

Send Input

Send all input for that tab window to selected other tab windows.

Scroll Up

Scroll the current tab window up one line.

Scroll Down

Scroll the current tab window down one line.

Page Up

Scroll the current tab window up one page.

Page Down

Scroll the current tab window down one page.

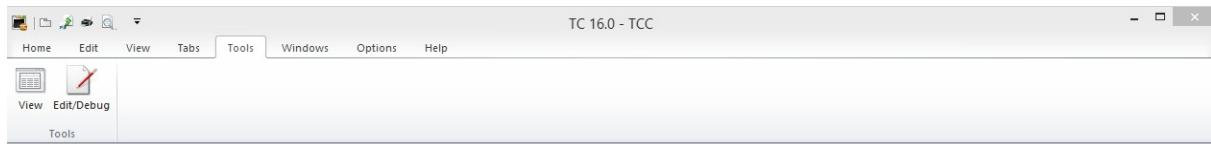
Home

Go to the beginning of the screen buffer for the current tab window.

End

Go to the end of the screen buffer for the current tab window.

3.2.1.1.6 Tools

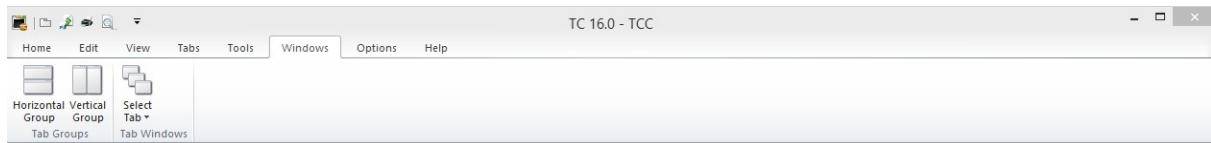
**View**

Starts the file viewer (VIEW in TCC).

Edit/Debug

The **Edit/Debug** button opens the **Take Command** batch debugger in a new **TCC** tab window.

3.2.1.1.7 Windows

**Horizontal Group**

Create a new tab bar by splitting the tabs window horizontally, and move the current tab window to the new tab bar.

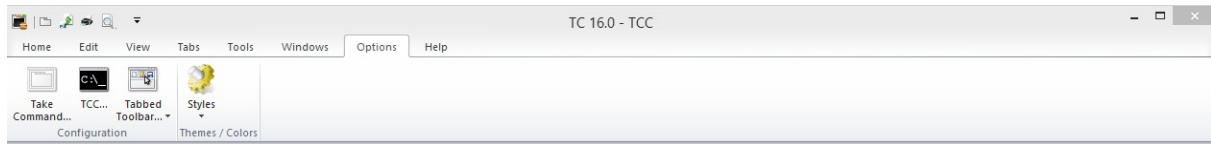
Vertical Group

Create a new tab bar by splitting the tabs window vertically, and move the current tab window to the new tab bar.

Select Tab

This menu will also display the names of all of the tabs, and allow you to switch to a new tab by selecting it in the menu.

3.2.1.1.8 Options

**Take Command**

Opens a [dialog](#) which you can use to change the **Take Command** configuration.

TCC

Opens a [dialog](#) which you can use to change the **TCC** configuration. This dialog does not affect any existing **TCC** windows, only new ones. If you want to change the configuration of an active **TCC** window, use the [**TCC OPTION**](#) command.

Tabbed Toolbar

Create a new toolbar button or tab, or save or reload the toolbar settings in TCMD.INI.

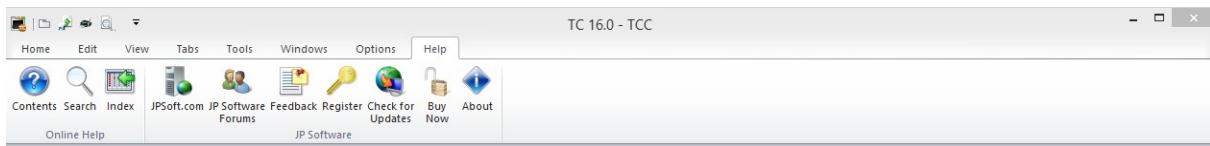
Styles

Select a predefined theme for **Take Command**. This will change the color and appearance of the **Take Command** window and its components (such as the tab window labels). You can also change the default font size.

The styles available are:

- Office 2007 Blue
- Office 2007 Silver
- Office 2007 Black
- Office 2007 Aqua
- Office 2007 System
- Office 2010 Blue
- Office 2010 Silver
- Office 2010 Black
- Office 2013
- Windows 7
- Windows 8

3.2.1.1.9 Help



See also: the [Help File](#) topic.

Contents

Displays the **Table of Contents** of the **Take Command** help file, from which you can directly navigate to any topic. This is the same display you will see if you select the **Contents** tab from within the help system.

Search

Displays the **Search dialog** of the **Take Command** help file, from which you can search for any topic. This is the same dialog you will see if you select the **Search** tab from within the help system.

Index

Displays the **Index dialog** of the **Take Command** help file, from which you can search for any keyword. This is the same dialog you will see if you select the **Index** tab from within the help system.

[Jpsoft.com/](#)

A hyperlink to the JP Software web site. Clicking it will attempt to display the JP Software home page in your default browser.

JP Software Forums

A hyperlink to the JP Software support forums.

Feedback

Leave a suggestion in the JP Software feedback forum.

Register

Enter an activation key to register Take Command.

Check for Updates

Query the JP Software web server to see if there is an updated version of **Take Command** available. If there is, the new version information will be displayed and you can choose to download and automatically update your existing version.

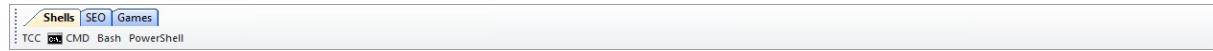
Buy Now

A hyperlink to our secure online store. Clicking it will attempt to display the store's first page in your default browser. Depending on your configuration, you may need to first establish an Internet connection.

About

Displays the **Take Command** version, copyright, and license information.

3.2.1.2 Tabbed Tool Bar



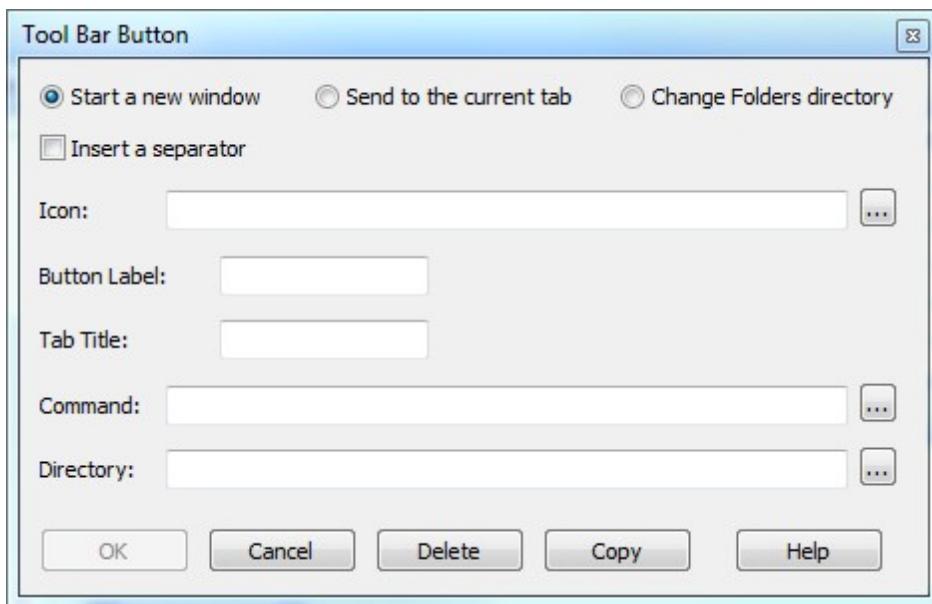
You can hide or show the toolbar with the **View / Tabbed Toolbar** menu entry, and you can detach and dock the toolbar by clicking on the left edge of the toolbar and dragging it to its new position. The new position of the toolbar will be saved when you close **Take Command** and restored when you restart.

The Tabbed Toolbar is an optional tabbed tool bar that you can use to execute internal or external commands, aliases, or batch files with the click of a mouse (or an accelerator key). You can define up to 20 tabs, each with up to 50 toolbar buttons. If you have only defined one tab, **Take Command** will hide the tab (to save screen space).

To create buttons for the tabbed toolbar, right click on the toolbar. You can modify or delete a button by right clicking on it. A dialog will appear to let you define the button label, tab title, command, and startup directory. There is a Copy option on the toolbar dialog which will duplicate the selected button so you can quickly create multiple variants of a command. You can also configure the tab toolbar from **TCC** with the [TCTOOLBAR](#) command.

You can also customize the Tabbed Toolbar by clicking on the button on the right side of the Quick Options menu and selecting **More Commands**, clicking on the Commands tab, and dragging icons to the toolbar.

You can reorder toolbar buttons by holding down the Alt key, pressing the left mouse button, and dragging a button to a new location.



3.2.1.3 Folders

The **Folders** window displays a tree view of the folders on your system. You can optionally select the folder by entering the name in the combo box on the [Explorer Toolbar](#).

You can detach, move and dock the **Folders** window by moving the mouse cursor to the upper left corner, left-clicking on the grabber bar, and holding the left mouse button down while dragging the window to its new location.

You can AutoHide the **Folders** window by clicking on the "push-pin" in the upper right corner, or selecting it from the drop-down menu. When in AutoHide, the window will be minimized to a single tab, and will automatically expand again when you move the mouse cursor over the tab.

You can completely hide the **Folders** and **List View** windows by toggling the **View / Folder and List Views** menu entry.

If the **Folders** and **List View** windows are disabled or hidden, **Take Command** will not try to load their content until the first time the window is displayed.

You can copy selected files in the **List View** to the clipboard with the Ctrl-C or Ctrl-Ins keys.

3.2.1.4 List View

The **List View** window displays the contents of the directory selected in the **Folders** window. You can choose the format (Large Icons, List, or Details) with the Views button on the [Explorer Toolbar](#).

You can optionally filter the contents of the **List View** window by entering a [regular expression](#) in the Filter combo box on the [Explorer Toolbar](#). You can also set the filter from **TCC** with the [TCFILTER](#) command.

You can detach, move and dock the **List View** window by moving the mouse cursor to the upper left corner, left-clicking on the grabber bar, and holding the left mouse button down while dragging the window to its new location.

You can AutoHide the **List View** window by clicking on the "push-pin" in the upper right corner, or

selecting it from the drop-down menu. When in AutoHide, the window will be minimized to a single tab, and will automatically expand again when you move the mouse cursor over the tab.

You can completely hide the **Folders** and **List View** windows by toggling the **View / Folder and List Views** menu entry.

If the **Folders** and **List View** windows are disabled or hidden, **Take Command** will not try to load their content until the first time the window is displayed.

You can copy selected files in the **List View** to the clipboard with the Ctrl-C or Ctrl-Ins keys.

3.2.1.5 Command Input

The **Command Input** window allows you to create and edit commands before sending them to the active tab window. This is not limited to entering something at the command prompt; you can feed strings anywhere a console app is expecting input. You can scroll back to previous lines, edit, and reexecute them. The **Command Input** window also has full undo/redo (up to 31 levels). You can automatically save the contents of the **Command Input** window when **Take Command** exits, and reload it when **Take Command** starts.

You can detach, move and dock the **Command Input** window by moving the mouse cursor to the upper left corner, left-clicking on the grabber bar, and holding the left mouse button down while dragging the window to its new location.

You can "AutoHide" the **Command Input** window by clicking on the "push-pin" in the upper right corner, or selecting it from the drop-down menu. When in AutoHide, the window will be minimized to a single tab, and will automatically expand again when you move the mouse cursor over the tab. You can completely hide the **Command Input** window by toggling the **View / Command Input** menu entry.

Each command line can contain up to a maximum of 65,535 characters.

The default directory for the **Command Input** window (used for filename completion) will be set to the selected directory in the **Folders** window.

You can use the following editing keys when you are entering a command (the words **Ctrl** and **Shift** mean to press the *Ctrl* or *Shift* key together with the other key named).

Cursor Movement Keys:

Left	Move the cursor left one character
Right	Move the cursor right one character
Ctrl-Left	Move the cursor left one word
Ctrl-Right	Move the cursor right one word
Home	Move the cursor to the beginning of the command
End	Move the cursor to the end of the command
PgUp	Scroll the Command Input window back one page
PgDn	Scroll the Command Input window forward one page

Insert and Delete Keys:

Ins	Toggle between insert and overstrike mode
Del	Delete the character under (or to the right of) the cursor, or the highlighted text
Bksp	Delete the character to the left of the cursor, or the highlighted text

Ctrl-L	Delete the word or partial word to the left of the cursor
Ctrl-R or Ctrl-Bksp	Delete the word or partial word to the right of the cursor
Ctrl-Home	Delete from the beginning of the line to the cursor
Ctrl-End	Delete from the cursor to the end of the line
Esc	Delete the entire line

Text Selection and Clipboard Cut/Copy/Delete/Paste:

Shift-Right	Highlight character right of cursor and move cursor
Shift-Left	Highlight character left of cursor and move cursor
Shift-Home	Highlight from cursor to beginning-of-line and move cursor
Shift-End	Highlight from cursor to end-of-line and move cursor
Ctrl-Shift-Right	Highlight word right of cursor and move cursor
Ctrl-Shift-Left	Highlight word left of cursor and move cursor
Ctrl-A	Select all
Ctrl-C	Copy highlighted text to the clipboard
Ctrl-V	Paste the first line of text from the clipboard at the current cursor position
Ctrl-X	Cut the selected text and copy it to the clipboard
Ctrl-Shift-Ins	Copy selected text and paste it at the current cursor position
Ctrl-B	Paste the last argument from the previous command line
Ctrl-0 to Ctrl-9	Paste the corresponding argument from the previous command line

Execution:

Ctrl-K	Save the current command line without executing it
Enter	Send the line to the active tab window
Ctrl-Enter	Send the line to the active tab window, but don't insert a CR/LF in the input window

Miscellaneous:

F1	Get help for the command (first argument on the line)
Ctrl-F1	Get help for the current word
Ctrl-Z	Undo last action
Ctrl-Y	Redo last undo
Tab or F7	Pop up the filename / sharename / variable completion window
Ctrl-E	Expand environment variables
Ctrl-.	Toggle between LFN and SFN
Ctrl-P	Print the command input window
Ctrl-T	Create new tab window

Mouse Clicks:

Single left click	Move text caret
Double left click	Select the current word
Triple left click (or single click in left margin)	Select the current line
Single right click	Context menu
Left click+drag	Drag current selection

To highlight text on the command line use the mouse or hold down the **Shift** key and use any of the cursor movement keys listed above. Once you have selected or highlighted text on the command line, any new text you type will replace the highlighted text. If you press **Bksp** or **Del** while there is text

highlighted on the command line, the highlighted text will be deleted.

3.2.1.6 Tab Windows

Take Command tab windows allow you to run multiple applications in their own tab inside a single **Take Command** session.

Although you can run any character-mode and many GUI applications in a tab window, the most common usage will be command processors or utilities. **Take Command** includes its own console-mode command processor (**TCC**, formerly known as **4NT**), but you can run any other command processor, including CMD, PowerShell, bash, etc. in a tab window. You can even tell **Take Command** to detect new console windows and automatically convert them to **Take Command** tab windows!

You can use the scroll bars or the **Alt** cursor keys to view text that has scrolled through the window. You can also save the contents of a tab window and scrollback buffer to a file, copy text from a tab Window to the clipboard, and copy text from the clipboard or from the tab window scrollback buffer to the command line. See [Highlighting and Copying Text](#) for information about saving and retrieving text in the tab window and [The Command Line](#) for complete details about using the **Take Command** console command line.

You can display the tabs on the top, left, right, or bottom of the **Take Command** window. You can also rotate the tab labels 90 degrees, allowing you to fit more labels without scrolling at the cost of slightly smaller tab windows. The small tab on the right of the tab window labels will open a new default tab window.

Take Command supports remapped console palettes in the tab windows. If you are running Windows Vista or later, Take Command will use the individual palette defined for each console.

Holding down the **Alt** key while spinning the mouse wheel will select tab windows.

Holding down the **Ctrl** key while scrolling the mouse wheel will change the font size in the **Take Command** console tab windows. Not all apps will be happy about you randomly changing their font size (and thus the console window size).

If the cursor is hidden in a console application running in a tab window, the tab window cursor will also be hidden.

You can also run simple GUI applications in tab windows. (This will not work for applications that have multiple parent windows.)

Splitter Windows

Take Command supports a splitter window option for each tab window. You must enable "Splitter Windows" in the Take Command configuration dialog (Tabs window), and restart **Take Command** to see the splitter.

To open a splitter window, drag the splitter button on the horizontal scrollbar to the right. The splitter window (on the right side) will not automatically scroll to the end when new output is displayed, or when you enter new commands. This allows you to scroll back in the screen buffer to review previous commands and output, and to select text from previous pages.

3.2.1.7 Status Bar

The **Take Command** window has a Status Bar that displays tooltips when you move the cursor over menu entries.



The status bar also displays the following information:

- ▶ The tab window size (columns x rows)
- ▶ The CPU usage (0 - 100%)
- ▶ The memory load (0 - 100%)
- ▶ The state of the Caps Lock key
- ▶ The state of the Num Lock key
- ▶ The state of the Scroll Lock key
- ▶ The current date
- ▶ The current time

You can hide the status bar fields by right clicking on the status bar and unchecking the fields you don't want to see.

There is a slider in the right corner that allows you to dynamically change the transparency level of the **Take Command** window. (You can also set the transparency in the **Options / Configure Take Command / Windows** dialog.)

If you are running **TCC** in a tab window and you enter an internal command, the status bar will display the brief syntax for that command. If you move the mouse over the status bar, a tooltip will pop up with the full syntax. If you enter an alias name, the status bar will identify it as an alias and display its value.

3.2.1.8 Keyboard Shortcuts

Take Command offers a number of keyboard shortcuts to change windows or select entries in the **Folders** and **List View** windows. If you are in a tab window, you need to set the **Left Alt Key** or **Right Alt Key** and **Left Ctrl Key** or **Right Ctrl Key** options in the **Take Command tab options** dialog. (Otherwise the keystroke will be sent to the console application rather than being interpreted by **Take Command**.)

All Windows

Alt-F4	Closes the Take Command window
Ctrl-Tab	Pops up a window allowing you to select the Folder , List View , or any of the tab windows.
Alt-F6	Cycle through the Folders View, List View, Command Input, and the active tab windows. If the Folder View and List View windows are disabled, Alt-F6 will toggle between the Command Input window and the active tab window.

Tab Window

Ctrl-F4	Closes the active tab window
Ctrl-T	Open a new tab
Ctrl-Ins	Copy the selected text to the clipboard
Ctrl-V	Paste the clipboard contents to the command line
Alt-Left	Change to the previous tab window
Alt-Right	Change to the next tab window
Alt-Up	Scroll tab window buffer up one line
Alt-Down	Scroll tab window buffer down one line
Alt-PgUp	Scroll tab window buffer up one page
Alt-PgDn	Scroll tab window buffer down one page
Alt-Home	Scroll to the beginning of the tab window buffer
Alt-End	Scroll to the end of the tab window buffer

The Tab Window keys can be changed by right-clicking on the **Take Command** menu, and selecting Customize / Keyboard.

Folders and/or View Windows

Alt-Enter	Display the Properties dialog for the selected folder or file
Ctrl-Enter	Open the selected folder or file
Shift-Enter	Explore the selected folder
Del	Delete the selected file(s) in the List View window to the recycle bin
Shift-Del	Permanently delete the selected file(s) in the List View window
F2	Rename the selected file(s) in the List View window
Ctrl-A	Select everything in the List View window

You can customize menu shortcut keys by right clicking on the **Take Command** menu bar and selecting "Customize / Keyboard".

3.2.1.9 Context Menus

Take Command displays a variety of context menus when you click on the right mouse button, depending on the location of the mouse cursor. If the mouse cursor is on the:

Toolbar

You can customize the tabbed toolbar.

Folders caption bar

You can detach, dock, AutoHide, or Hide the **Folders** window.

Folders view

Displays the Windows Shell context menu.

List View caption bar

You can detach, dock, AutoHide, or Hide the **List View** window.

List View

Displays the Windows Shell context menu.

Tab Bar

You can create a new tab, run an application (in a tab or stand-alone) or move an existing console window to a **Take Command** tab window.

Tab Labels

You can attach, detach, rename, and close tabs;

Tab Windows

You can copy selected text to the clipboard or paste the clipboard contents to the application in the tab window. (Paste works by sending the contents of the clipboard to the tab window as a series of keystrokes, so it will work with any application that accepts input.)

3.2.1.10 Using the Scrollback Buffer

Take Command retains the text displayed on its tab windows in a "scrollback buffer". You can scroll through this buffer using the mouse and the vertical scroll bar at the right side of the **Take Command** window, just as you can in any Windows application. You can also use the **Alt-Up** and **Alt-Down** keys to scroll the display one line at a time from the keyboard, and the **Alt-PgUp** and **Alt-PgDn** keys to scroll one page at a time.

If you scroll back through the buffer to view previous output, and then enter text on the command line, **Take Command** will automatically return to the bottom of the buffer to display the text.

You can set the size of the scrollback buffer on the [Tabs](#) tab of the [configuration dialogs](#).

3.2.1.11 Highlighting & Copying Text

While you are working at the **Take Command Console** prompt you can use common Windows keystrokes to edit commands, and use the Windows clipboard to copy text between **Take Command** and other applications. You can also select all of the text in a **Take Command** tab window buffer by using the **Select All** command on the Edit menu.

The right mouse button will pop up an **Edit** context menu.

Take Command allows both line and column selection. If you hold down the **Ctrl** key while dragging the mouse, **Take Command** will select a rectangular block of text. Otherwise, as you drag the mouse down **Take Command** will highlight text to the end of the previous line.

To copy text from a **Take Command** tab window to the clipboard, first use the mouse to highlight the text, then right click and select **Copy**, or use the **COPY** command on the [Edit menu](#). You can optionally combine multiple selected lines into a single line before placing it in the clipboard by holding down the **Ctrl** key and selecting **Copy** (or **Copy+Paste** or **Copy+Paste+Run**) from the right-click context menu or the Edit menu. If you have an existing multiline selection in the clipboard, you can copy it to a single line (with the CR/LF's replaced by a space) in the **Take Command** window by holding down the **Ctrl** key and selecting **Paste**. (If you hold down the **Ctrl** key and the selection wraps around the last screen column, the lines will be appended without an intervening space.)

If you double-click on a word in the **Take Command** window, the entire word is highlighted or selected. If you triple click, the entire line is selected.

To highlight text on the command line use the **Shift** key in conjunction with the **Left**, **Right**, **Ctrl-Left**, **Ctrl-Right**, **Home**, and **End** cursor movement keys. The **Del** key will delete any highlighted text on the command line, or you can type new text to replace the highlighted text.

While **Take Command** tab windows contain text, they are not document windows like those used by word processors and other similar software, and you cannot move the cursor throughout the window as you can in text processing programs. As a result, you cannot use the Windows shortcut keys like **Shift-Left** or **Shift-Right** to highlight text in the window. These keys work only at the command line; to highlight text elsewhere in the window you must use the mouse.

To copy text from the clipboard to the command line use **Ctrl-V**, or the Paste command on the Edit menu.

To paste text from elsewhere in a **Take Command** tab window directly onto the command line, highlight the text with the mouse and press **Ctrl-Shift-Ins**, or use the **Copy+Paste** command on the Edit menu. This is equivalent to highlighting the text and pressing **Ctrl-Ins** followed by **Ctrl-V**, except that it will not change the contents of the clipboard. It's a convenient way to copy a filename from a previous DIR or other command directly to the command line.

You should use caution when pasting text containing carriage return or line feed characters onto the command line. If the text you insert contains one of these characters the command line will be executed just as if you had pressed Enter. If you insert multiple lines, the text will be treated just like multiple lines of commands typed at the prompt.

You can also use Windows' [Drag and Drop](#) facility to paste a filename from another application onto the command line, and you can access the clipboard with [redirection](#) to or from the CLIP: device, or with the [@CLIP](#) variable function.

3.2.1.12 Resizing the Window

You can resize the **Take Command** window at any time by dragging a corner with the mouse. Resizing the window changes the number of rows and columns of text which will fit in the command window (the actual number of rows and columns for any given window size depends on the font you are using). **Take Command** reacts to these changes using two sets of rules: one for the height and one for the width.

When the height of the command window changes, future commands simply use the new height as you see it on the screen. For example, if you reduce the window to three rows high and do a [DIR /P](#) (display a directory of files and pause at the bottom of each visual "page"), DIR will display two lines of output, a prompt ("Press any key to continue ..."), and then pause. If you expand the window to 40 lines high and repeat the same command, DIR will display 39 lines, a prompt, and then pause.

However, when the width of the window changes, **Take Command** must check the current virtual screen width. The virtual width is the maximum number of characters on each line in **Take Command**'s internal screen buffer. You can think of it as the width of the data which can be displayed in the **Take Command** window, including an invisible portion to the right of the window's right-hand edge. When the virtual width is larger than the actual width, a standard horizontal scroll bar is displayed to allow you to see any hidden output.

The [ROWS](#) internal variable can be used to determine the current screen height.

The virtual screen width starts at 80 columns or the number of columns which fit into the startup **Take Command** window, whichever is larger. The [_COLUMNS](#) internal variable can be used to determine the current virtual screen width.

If you expand the **Take Command** window beyond its previous virtual width, the virtual width is automatically increased. This ensures that the internal buffer can hold lines which will fill the newly enlarged window. If you shrink the window, the virtual width is not reduced because this might require removing output already on the screen or in the scrollback buffer.

As a result, widening the window will make future commands use the new enlarged size (for example, as the window is widened DIR /W, which displays a "wide" directory listing, will display additional columns of file names). However, if the window is narrowed future commands will still remember the enlarged virtual width, and display data to the right of the window edge. Use the horizontal scroll bar to make this data visible.

When the font is changed, **Take Command** will recalculate the virtual screen width.

3.2.1.13 Drag & Drop

Take Command is compatible with the **Drag-and-Drop** facility in Windows.

To add a filename to the command line using drag and drop simply drag the file from an application (or the **Take Command** folder or list view windows), and drop it anywhere inside a **Take Command** tab window. The full name of the file will be pasted at the current cursor position.

Take Command is a drag and drop "client", which means it can accept files dragged in from other applications and paste their names onto the command line as described above. It is not a drag and drop "server", so you cannot drag filenames from the **Take Command** tab windows into other applications. However you can copy filenames and other text from the **Take Command** window to other applications using the clipboard; see [Highlighting and Copying Text](#) for details.

3.2.1.14 Take Command Dialogs

The **Take Command** menus lead to several dialog boxes. Each is listed here for quick reference, though in general you will find it easier to learn about each one from the context in which it is used (for example, the information referenced below on the tool bar dialog will be more useful after you have read the section on the [tool bar](#)).

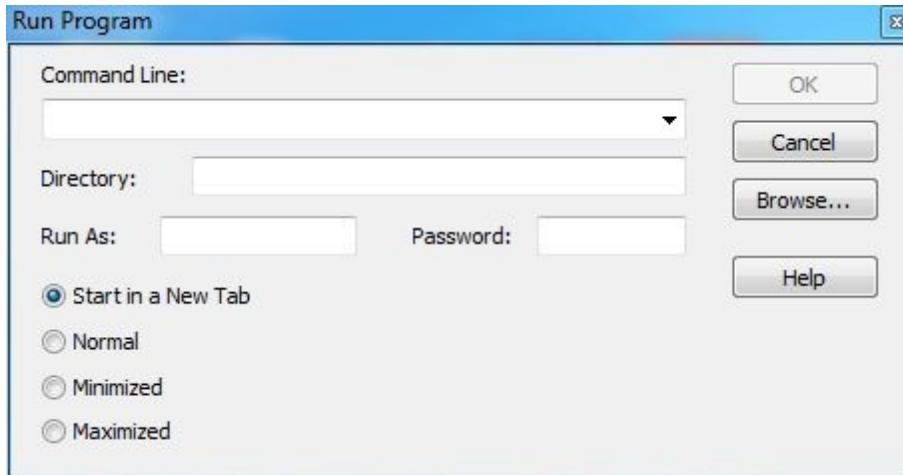
Take Command uses standard Windows dialogs for tasks like printing, selecting a font, or browsing files and directories. Since these dialogs are provided by Windows, not **Take Command**, and are common to many different Windows programs; they are not documented within this help system.

The reference in parentheses after certain dialogs listed below shows the name of the [menu](#) you can use to access that dialog.

Run Program	(Tabs)
Find Files	(Edit)
Configure Take Command	(Options)
Windows Tab	
Tabs Tab	
AdvancedTab	
Configure Tab Toolbar	(Options)

3.2.1.14.1 Run Program

The Run Program dialog, started from the [Tabs menu](#), allows you to run a program by typing its name or browsing the disk.



In the **Command Line** edit box, you can enter the name of any executable program plus command line parameters. If you click on the arrow to the right of the edit box, the dialog displays a list of previous commands you have entered during the current **Take Command** session.

The **Browse** button leads to a standard file browser from which you can select any executable program. Your choice will be placed in the Command Line edit box, and you can add parameters before selecting OK to run the program.

You can enter an optional startup directory in the **Directory** edit box.

You can specify an optional user name and password in whose context the program should be run. (Depending on your user privileges, you may not be able to run the program in a **Take Command** tab window.)

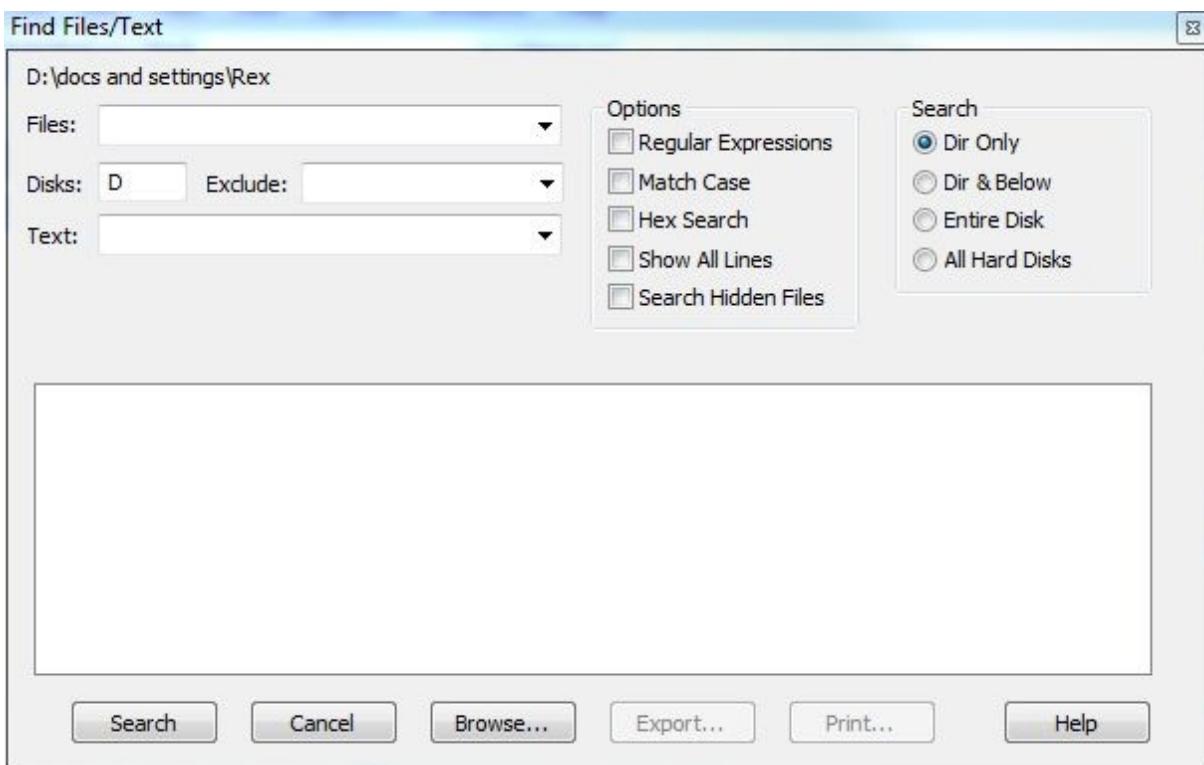
Start in a New Tab will start the application in a new **Take Command** tab window. This will usually be a Windows console (character-mode) application, but **Take Command** can also run many simple GUI applications in a tab window (provided the application does not have multiple parent windows).

The **Normal**, **Minimized**, and **Maximized** buttons determine the type of window that will be used for the program. If you select Minimized, the program will start as an icon on the Taskbar. Maximized starts the program in a full-screen window. The Normal button lets the operating system select the size and position of the program's window.

If you're running Windows Vista or later, you can opt to **Run as Administrator** to start the program with admin privileges. (This option will invoke the Windows UAC dialog).

3.2.1.14.2 Find Files

The Find Files/Text dialog box, available from the [Edit menu](#), gives you the same features as the [FFIND](#) command, in dialog form.



Enter the file name or names you wish search in the **Files** field. You can use [wildcards](#) and [include lists](#) as part of the file name. To select files from previous searches in the same **Take Command** session, click on the down arrow beside the **Files** field, or press the up or down arrow while the input cursor is in the **Files** field. You can also use the **Browse** button to find files to include in the search.

Enter the drive(s) you want to search in the **Disks** field. This field is ignored unless **Entire Disk** is selected in the **Search** portion of the dialog. If you select **All Hard Disks**, this field is set automatically to include all hard disk drive letters **Take Command** finds on your system.

If you use wildcards to specify the files to search, you can narrow the search with the **Exclude** field by specifying files that you want to exclude from the search. Like the **Files** field, the **Exclude** field can contain [wildcards](#) and [include lists](#). For example, if you want to search all files with an extension beginning with **i** except for **.ICO** and **.INI** files, you could enter ***.i*** in the **Files** field and then ***.ico;*.ini** in the **Exclude** field.

Enter the text (or hexadecimal values) you are searching for in the **Text** field. You can use [extended wildcards](#) in the search string to increase the flexibility of the search. Use back-quotes ` around the text if you want to search for characters which would otherwise be interpreted as wildcards. For example, to search for an **A**, followed by some number of other characters, followed by a **B**, enter the **A*B** as your search string. To search for the literal string **A*B** (**A**, followed by an asterisk, followed by **B**), enter **`A*B`** as your search string search string (the closing back-quote is optional).

You can search either using the **TCC**-style extended [wildcards](#) or with [regular expressions](#).

The **Match Case** box, when it is selected, makes the search case-sensitive. This option is ignored if **Hex Search** is selected. The **Hex Search** option signals that you are searching for hexadecimal values, not ASCII characters. See the [FFIND](#) command for more details.

If you enable **All Lines**, every matching line from every file will be displayed; otherwise only the first matching line from each file will be displayed. Unless you enable the **Hidden Files** option, files with the hidden and system [attributes](#) will not be included in the search.

The radio buttons in the **Search** area let you specify where you want **Take Command** to look for files. If you select **Dir Only** or **Dir & Below**, the search will begin in the current default directory, shown above the **Files** box. If you select **Entire Disk**, **Take Command** will use the drives that you specified in the **Disks** field. If you select **All Hard Disks**, **Take Command** will search all the hard disk drives it finds on your system.

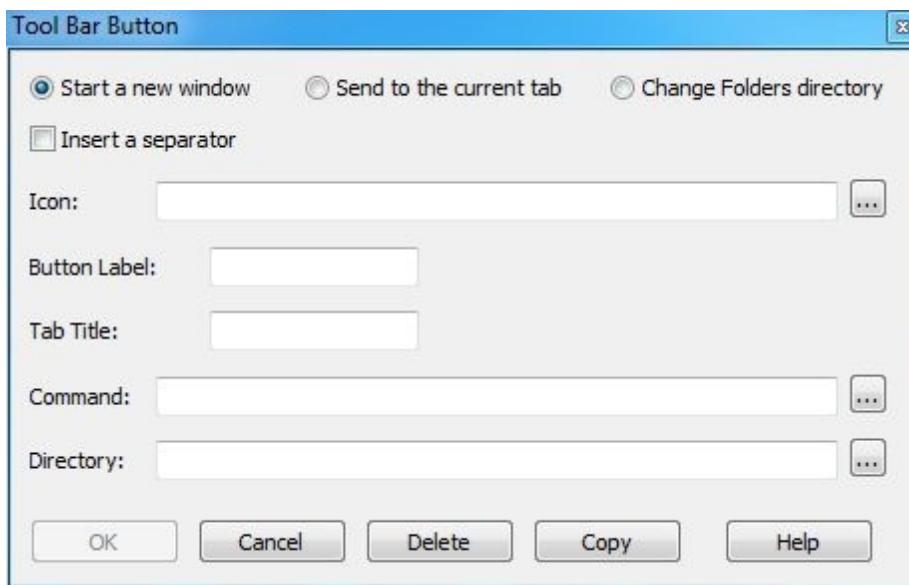
To start the search, press the **Search** button. Once the search has started the **Search** button changes to a **Stop** button, which you can use to interrupt the search before it is finished.

If the search yields a list of matching files, you can save that list with the **Export** button or send it to the default printer with the **Print** button.

If you select one of the matching files in the list (by double-clicking on it, or selecting it with the cursor and pressing Enter), **Take Command** will display another dialog with complete directory information about the file. From that dialog you can **Run** the file (if it is an executable file, a batch file, or has an [executable extension](#)), or **Edit** the file (with its associated app, or Notepad if there is no association). When you exit from the editor, the original list of matching files will still be available.

3.2.1.14.3 Tab Window Toolbar

You can create new buttons by right clicking on an empty part of the toolbar and selecting "Add Button" from the popup menu. You can modify or delete a button by right clicking on it. A dialog will open to let you define the button label, tab title, command, and startup directory.



A toolbar button can either open a new tab, send keystrokes to the current tab, or change the directory displayed in the Folder and List views.

You can define a toolbar button to display either an icon, a text label, or both. You must specify either the **Icon** or **Label** fields. If you enter both, **Take Command** will display the text to the right of the icon on the button.

In the **Icon** field, enter the filename for the icon (.ico) that you want to display on the button. If you specify an .exe filename, **Take Command** will use the first icon in that file. You can use the **Browse** button to find the file.

In the **Button Label** field, enter the text that you want to display on the button.

In the **Tab Title** field, enter the text that you want to display on the tab window title. This field is optional, and is only used when creating a new tab window.

In the **Command** field, you can enter either the command to be started in a new window ("Start a new window"), or the keystrokes to be sent to the current tab ("Send to current tab"). You can use the **Browse** button to find a file to be entered at the beginning of the Command field. If the command is a GUI app, it will be started in a new window outside of **Take Command**. If it is a console app, it will be started in a new tab window. (You can start GUI apps in a **Take Command** tab window by using the [START /TAB](#) option.)

If the tab is sending keystrokes to the current tab, the text is in the same format as the [KEYSTACK](#) command in the **Take Command Console**:

If you're starting a new window, the **Directory** field will set the startup directory for the command. If you are changing the Folders directory, the **Directory** field specifies the new directory. You can use the **Browse** button to find the directory.

The **command** and **directory** fields can include environment variables and **TCC** internal variables and variable functions. Note that the variable expansion occurs in **Take Command**, not **TCC**, so internal variables like [%_ cwd](#) will not probably work as expected.

Keystroke Interpretation

Characters entered within double quotes, e.g., "**abc**" will be sent to the active console application as is. The only items allowed outside the quotes are key names, the ! and /W options, and a repeat count.

If **keyname** is a number, it is interpreted as an ASCII character value.

Repetition. To send **keyname** several times, follow it with a space, left bracket [, the repetition count, and a right bracket]. For example, the command below will send the Enter key 4 times:

enter [4]

The repeat count works only with an individual **keyname**. It cannot be used with quoted strings. You must have a blank space between the **keyname** and the repetition count.

If you exit by choosing the **OK** button, any changes you have made will be saved in TCMD.INI, and reloaded automatically the next time you start **Take Command**. If you use the **Cancel** button, your changes will be discarded.

The tool bar can also be configured with the [TCTOOLBAR](#) command.

3.3 Configuration Options

Take Command offers a wide range of configuration options, allowing you to customize their operation for your needs and preferences. The **Take Command** menu entry **Options / Configure Take Command** invokes the [Take Command Configuration Dialog](#).

3.3.1 Initialization (.INI) Files

Part of the power of **Take Command** is its flexibility, in allowing you to alter its configuration to match your style of computing. **Take Command**'s configuration is controlled through a file of initialization information.

See [Locating the .INI files](#) below to find out how **Take Command** locates its *TCMD.INI* file.

Modifying the TCMD.INI File

You can create, add to, and modify the *TCMD.INI* file with the **Configure Take Command** selection on the [Options menu](#), or (for the **TCC**-specific sections) with the [configuration dialog](#), available via the [OPTION](#) command.

Most of the changes you make in the [configuration dialog](#) take effect immediately. A few (e.g., startup tabs and buffer size) only take effect when you start a new **Take Command** session. See the online help for each individual dialog page if you are not sure when a change will take effect.

Take Command reads its *TCMD.INI* file (see [Locating the .INI file](#)) when it starts, and configures itself accordingly. The *.INI* file is not reread when you change it manually. For manual changes to take effect, you must restart **Take Command**.

Each item that you can include in the *.INI* file has a default value. You only need to include entries in the file for settings that you want to change from their default values.

Using the TCMD.INI File

Some settings in the *.INI* file are initialized when you install **Take Command**; others are modified as you use and when you exit **Take Command**.

You can optionally include environment variables in the *TCMD.INI* [4NT] and [TCMD] sections; they will be expanded when *TCMD.INI* is loaded. If you want to delay expansion until command execution time (for example, with ColorDir) you will need to double the %'s.

Locating the TCMD.INI File

When starting **Take Command** or a **Take Command Console (TCC)** shell:

- ▶ If there is an `@d:\path\inifile` option on the startup command line, **Take Command** will use the path and file name specified there.
- ▶ Otherwise, the default *TCMD.INI* file name is used, and the search starts in the directory where the **Take Command** program file is stored. If the *.INI* file is not found, **Take Command** will look in the `%LOCALAPPDATA%` directory.

If no *.INI* file is found, all options are set to their default values. A new *.INI* file will be created, using the default location and name, as explained above.

TCMD.INI File Sections

The *TCMD.INI* file has a number of sections. Each section is identified by the section name in square brackets on a line by itself. **Take Command** stores the user-defined options in [**TakeCommand**]; **TCC** stores its user-defined options in [**4NT**].

3.3.2 Take Command Configuration Dialog

This dialog, available from the [Options menu](#) in **Take Command**, contains four pages or "tabs" of options that let you change the way **Take Command** looks and works.

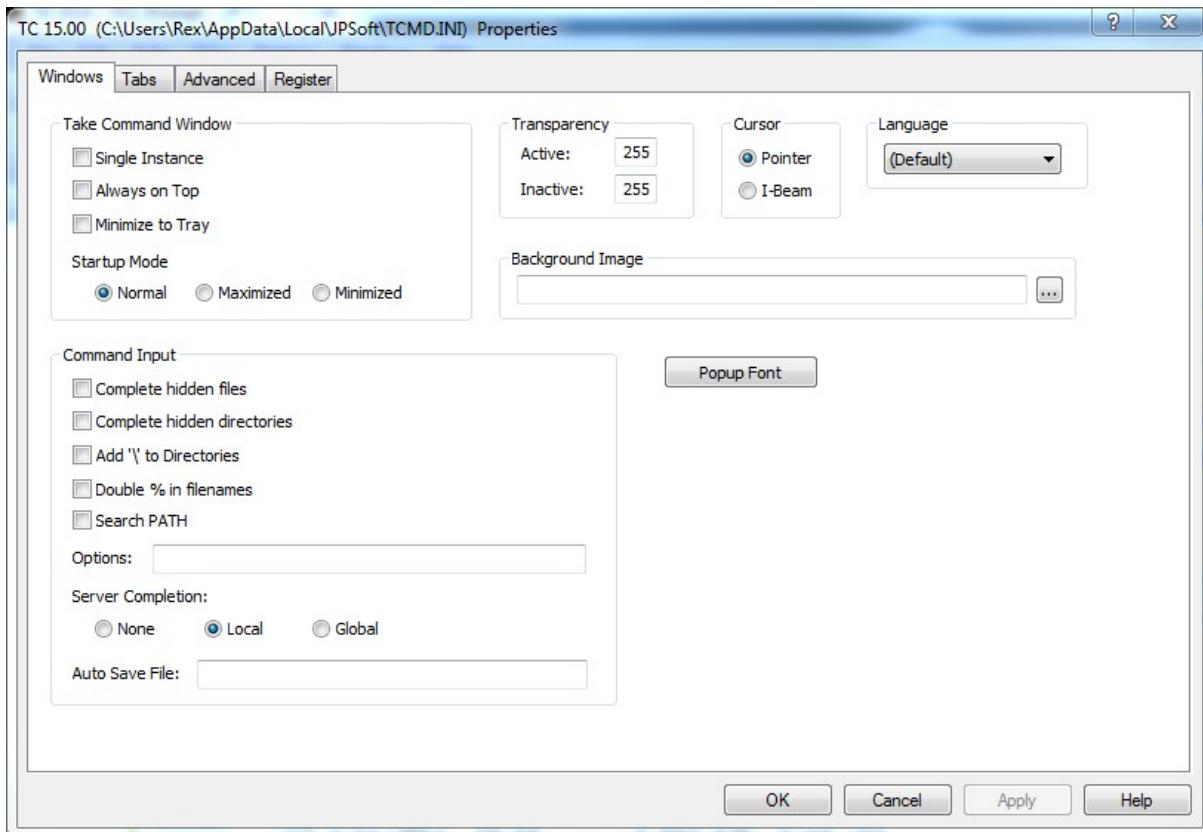
The configuration dialog displays the name of the active **TCMD.INI** file in the title bar.

Unless you select the **Cancel** button, any changes you make will take effect immediately. If you select **Apply**, the settings will only apply for the duration of that session. If you select **OK**, the settings will be recorded in the appropriate section of the [TCMD.INI file](#) and will be in effect each time you start **Take Command**.

While you are using the dialog, you can move between sets of configuration options by clicking on the individual tabs. The options available in this dialog are:

[Windows](#)
[Tabs](#)
[Advanced](#)
[Registration](#)

3.3.2.1 Windows



If you are not familiar with the purpose or use of the Windows configuration dialog, review the main [configuration dialogs](#) topic before continuing.

Take Command Window

Single Instance restricts **Take Command** to a single instance. (If you try to start another copy of **Take Command**, the previous instance will be brought to the foreground.)

Always on Top makes **Take Command** a topmost window (i.e., it will remain on top of all other non-topmost windows).

Minimize to Tray minimizes **Take Command** to the system tray instead of the task bar.

Startup Mode - The **Normal**, **Max**, and **Min** buttons select the initial state for the **Take Command** windows.

Command Input:

These options affect filename completion (tab or F7) in the optional [Command Input](#) window.

Complete hidden files : If enabled, hidden and system files will be displayed.

Complete hidden directories : If enabled, hidden directories will be displayed.

Add '\' to Directories : If enabled, a \ (backslash) is automatically appended to directory names (or / to FTP directories).

Double % in filename : If enabled, and the filename has embedded % characters, and the first argument on the command line is an internal command, the % characters will be doubled so that variable expansion won't delete (or unexpectedly expand) the filename. (This will not affect filenames on lines beginning with aliases or variables.)

Search PATH : If enabled, the directories in the [PATH](#) variable are searched if a match isn't found in the current directory.

Options : Sets the files returned during filename completion for selected commands. The format is the same as that used by the [TCC FILECOMPLETION](#) environment variable. See [Customizing Filename Completion](#) for a detailed explanation of selective filename completion.

Server Completion : Configures server name completion (see [Filename Completion](#) for information on how to use server name completion). **Local** lists only local servers (i.e., those in your "network neighborhood"). **Global** will enumerate the entire network. **None** will disable server completion; this may be necessary to prevent "hanging" if you start typing a server name and accidentally press Tab, and your local domain is very large or slow to respond.

Auto Save File: The name of a file used to save and restore the Command Input window when **Take Command** exits and starts.

Transparency:

Transparency sets the transparency level for the **Take Command** window. The range is from 20 (nearly invisible) to 255 (completely opaque). You can define both the active transparency (when **Take Command** is the foreground window, and the inactive transparency (when **Take Command** is in the background).

Cursor:

The **Pointer** and **I-Beam** buttons let you select the type of cursor which **Take Command** will use in

the tab windows.

Language:

The **Language** combo box allows you to override the default language that **Take Command** uses for menus and dialogs.

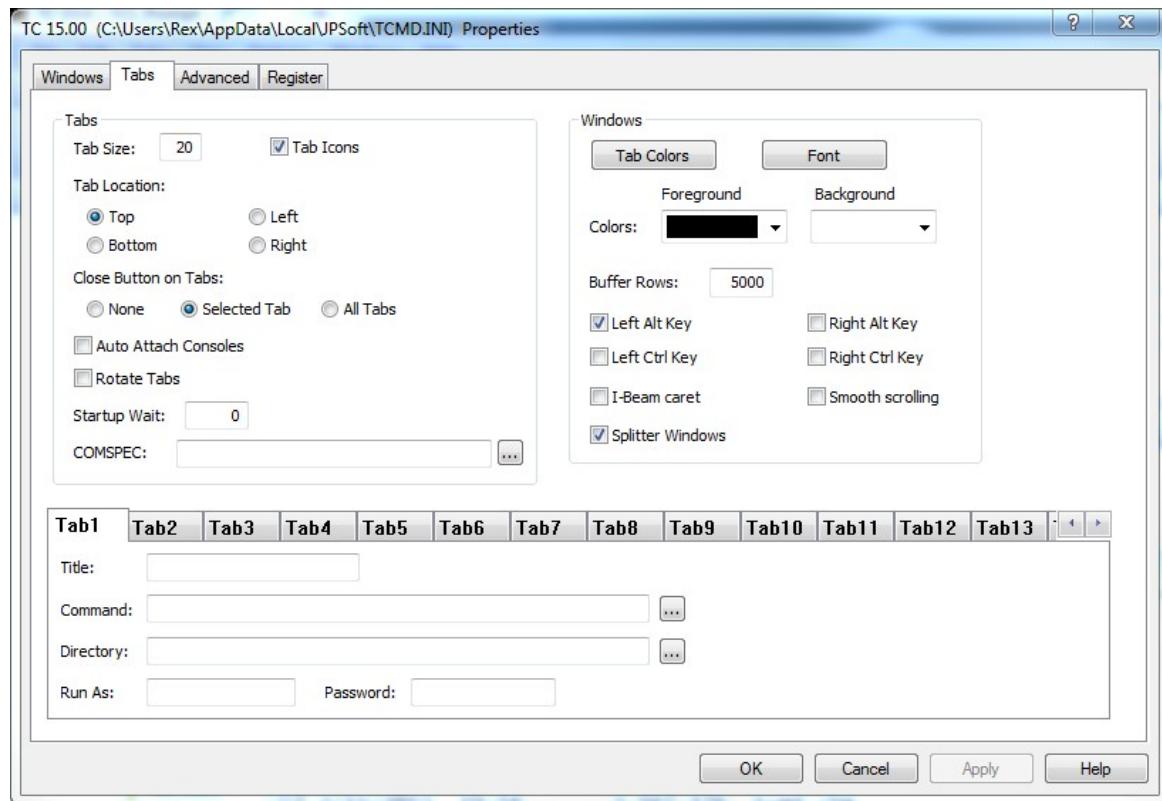
Background Image:

A bitmap file (.BMP) to use as the background in **Take Command** tab windows. (Changing this option will not affect existing windows, only new ones.)

Pop-Up Font:

Set the font to use in the filename completion popup window.

3.3.2.2 Tabs



If you are not familiar with the purpose or use of the Tab Windows configuration dialog, review the main [configuration dialogs](#) topic before continuing.

Tabs:

Tab Size sets the maximum length of the tab label text (in characters).

Tab Icons displays the application's icon in the tab label.

You can position the tab labels with the **Top**, **Bottom**, **Left**, and **Right** options.

Close Tab Button defines the close button (X) on the tab labels. You must restart **Take Command** for this option to take effect.

Auto Attach Consoles converts all new console windows to **Take Command** tab windows.

Rotate Tabs rotates the tab labels 90 degrees. This allows you to fit many more tabs in the window, at the cost of a reduced window size.

Startup Wait is the number of milliseconds to wait before loading the next startup tab. The range is 0 - 5000. (This should only be needed in rare cases, when the apps are accessing the same files and are interfering with one another while starting.)

COMSPEC sets the program you want to start in new tabs. If no COMSPEC option is set, **Take Command** will run its default command processor (**TCC**).

Windows:

Tab Colors lets you define a custom color palette to use in the tab windows. **Take Command** will first try to retrieve a color palette from the console (Vista or later only). If the console app is using a custom palette, **Take Command** will use that palette for the tab window. If there is no custom palette defined for this console, **Take Command** will use the palette set by **Tab Colors**.

The **Font** button opens a standard Windows font dialog that lets you select the font, point size, and font style for **Take Command** tab windows.

Colors lets you select the default foreground and background colors for text in the **Take Command** tab windows. (These colors will be overwritten if the application in the tab window sets its own colors.)

Buffer Rows sets the number of rows in the console-mode screen buffer.

Left Alt Key sends the left Alt key to **Take Command**, so you can use it to invoke the **Take Command** menu or scroll the window. (The right Alt key will be passed to the application in the tab window.)

Right Alt Key sends the right Alt key to **Take Command**, so you can use it to invoke the **Take Command** menu or scroll the window. (The left Alt key will be passed to the application in the tab window.)

Left Ctrl Key sends the left Ctrl key to **Take Command**. (The right Ctrl key will be passed to the application in the tab window.)

Right Ctrl Key sends the right Ctrl key to **Take Command**. (The left Ctrl key will be passed to the application in the tab window.)

I-Beam Caret uses the I-Beam (vertical) caret in tab windows instead of the console underline (horizontal) caret.

Smooth Scrolling scrolls the **Take Command** tab windows smoothly (2 pixels at a time instead of a line at a time). This will slow down scrolling and output from fast programs. The smooth scrolling is only visible if the underlying console window is scrolling two lines or less, so if the tab window is

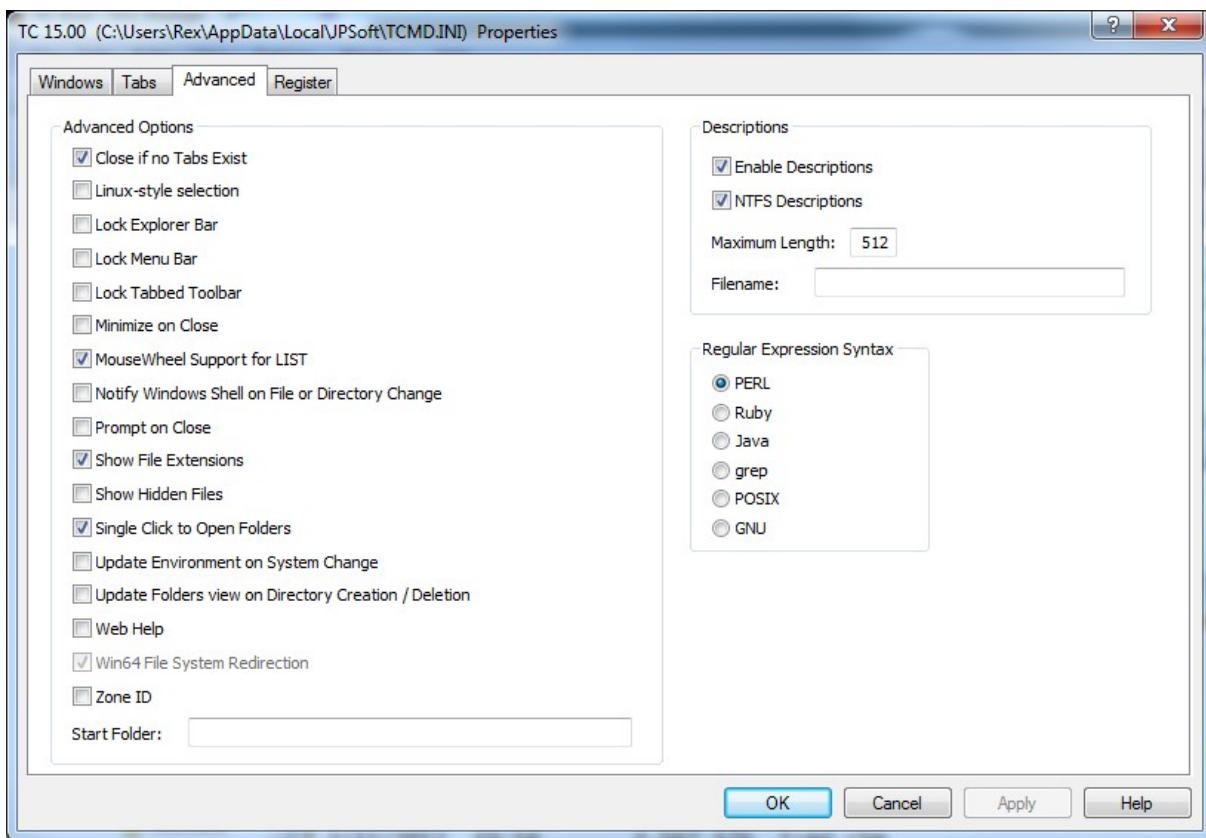
displaying a lot of fast output you probably won't see it.

Splitter Windows enables the splitter option on the horizontal scrollbar in tab windows.

Startup Tabs:

The **Startup Tabs** specify programs to run in each tab at startup. You can specify an optional title, the command line (the name of the program and any optional arguments), an optional startup directory, and optionally the user context where the tab should run. You can define up to 25 startup tabs.

3.3.2.3 Advanced



If you are not familiar with the purpose or use of the Advanced Directives configuration dialog, review the main [configuration dialogs](#) topic before continuing.

Advanced Options:

Close if No Tabs Exist : If disabled, **Take Command** won't close if there are no tab windows open.

Linux-style Selection : Copy selected text automatically to the clipboard when the left mouse button is released.

Lock Explorer Bar : Locks the Explorer toolbar in place so it cannot be moved or docked. (Takes effect after restarting **Take Command**.)

Lock Menu Bar : Locks the menu in place so it cannot be moved or docked. (Takes effect after

restarting **Take Command**.)

Lock Tabbed Toolbar : Locks the tabbed toolbar in place so it cannot be moved or docked.
(Takes effect after restarting **Take Command**.)

Minimize on Close : If enabled, clicking on the Close button or pressing Alt-F4 will minimize **Take Command** instead of closing it. To close **Take Command**, select File / Exit.

MouseWheel Support for LIST : If enabled, **Take Command** will pass the mouse wheel messages to **TCC** when it detects that **TCC** is executing its internal LIST command.

Notify Windows Shell on File or Directory Change : Notify the system shell when changing files or directories.

Prompt on Close : **Take Command** will pop up a confirmation message box when exiting.

Show File Extensions : If enabled, **List View** will always show file extensions even if they are disabled in the Windows Shell folder properties.

Show Hidden Files : If enabled, the **Folders** and **List View** windows will show hidden files. (You must restart **Take Command** after changing this option.)

Single Click to Open Folders : Single click to open an item in the **Folders** window. (This will also close any other expanded items.) If this option is not enabled, **Take Command** will open folders in the **Folders** window with a double click and will not close other expanded items. You must restart **Take Command** after changing this option.

Update Environment on System Change : If enabled, **Take Command** will monitor the WM_SETTINGCHANGE message and if the environment is specified, update the environment from the User, Volatile, and System registry entries. The updates are done whenever **Take Command** displays a prompt (to prevent the environment from changing in the middle of a batch file).

Update Folder View on Directory Creation / Deletion : If enabled, **Take Command** will monitor the system for new, deleted, or renamed directories and automatically update the Folder view window.

Web Help : If enabled, **Take Command** will use the browser-based help (at <http://jpsoft.com/help/index.htm>) instead of the local help. Using web help allows you to add comments to the help topics.

Win64 File System Redirection : If disabled, overrides the default Win64 behavior of remapping windows\system32 calls to windows\SysWOW64.

Zone ID : Set the NTFS Zone ID security when running executables downloaded from the Internet.
(Note that CMD never checks for the Zone ID, so setting it may introduce a minor incompatibility.)

Start Folder : The initial directory to display in the **Folders / List View** windows.

Descriptions:

Enable Descriptions : Set description display for the List View window.

NTFS Descriptions : If set, **Take Command** uses the Comments field in the NTFS SummaryInformation stream for each file to hold its description, instead of the DESCRIPT.ION file. The advantages are that the description will always remain with the file regardless of what program copies, moves, or renames it. The disadvantage is that you cannot attach a description to directories.

Maximum Length : Set the description length limit. The allowable range is 20 to 512 characters.

Filename : Sets the file name in which to store file descriptions. The default file name is DESCRIPT.ION.

Regular Expression Syntax:

Sets the [regular expression](#) syntax.

3.3.2.4 Register

There are no separate **trial** and **registered** versions of our products. Without registration, a trial version is fully functional for 30 days of use.

The Register tab allows you to register **Take Command**. When you purchase a new or upgrade copy of **Take Command**, you will receive an email with your name and registration key. Enter the registration information exactly as you received it in the email (preferably by copying & pasting with the clipboard). Remember to save your registration key in a safe place in case you need to reinstall. If you have lost your registration key, you can request a replacement by contacting JP Software at support@jpsoft.com.

When you enter your registration information, **Take Command** will save it in the Windows Registry.

(Windows 7, Windows 8 and Vista) If you are running an elevated administrator session, you can optionally check the "Register for All Users" option.

3.3.3 Windows Explorer Integration

Take Command includes five batch files to help integrate **Take Command** and **TCC** with Windows Explorer:

TCCBATCH.BTM	Batch file to make TCC the default handler (replacing CMD.EXE) in Windows Explorer for .BAT, .BTM, and/or .CMD batch files.
TCCHERE.BTM	Batch file to add a "TCC prompt here" menu entry to Explorer. Clicking on this will open a new copy of TCC in the selected directory.
TCCTABHERE.BTM	Batch file to create a "TCC tab window here" in the Take Command Folders and List View context menus.
TCMDBATCH.BTM	Batch file to make Take Command the default handler (replacing CMD.EXE) in Windows Explorer for .BAT, .BTM, and/or .CMD batch files
TCMDHERE.BTM	Batch file to add "Take Command Prompt Here" menu entry to Windows Explorer. Clicking on this will open a new copy of TCC in the selected directory.

3.4 Take Command and TCC Integration

Take Command and **TCC** are tightly integrated and pass messages and commands back and forth. (If you are running another application, such as CMD or PowerShell in a **Take Command** tab window, you will not have access to these commands and variables.)

The "Change Folder" combobox on the **Take Command** toolbar recognizes **TCC** [directory aliases](#) (if you are using a global alias list).

Internal commands:

[CDD](#) /T or /TO - Change the selected folder in the **Take Command** Folders window.

[START](#) /TAB - Start the process in a new **Take Command** tab window.

[TCFILTER](#) - Display or set the filter for the **Take Command** list view.

[TCTOOLBAR](#) - Change the tool bar buttons.

[WINDOW](#) - When run in a tab window, the WINDOW options act on the **Take Command** window, not the **TCC** tab window.

Internal Variables:

[TCFILTER](#) - returns the **Take Command** list view filter.

[TCFOLDER](#) - returns the selected folder in the Folders window if in a **Take Command** tab window.

[TCTAB](#) - returns 1 if TCC is in a **Take Command** tab window.

Take Command also creates two environment variables that can be queried by its child tab window processes:

[TCMD](#) - the full pathname of the **Take Command** executable

[TCMDVER](#) - the version & build number (i.e., 16.00.20).

Syntax Messages:

Take Command will display the syntax for **TCC** internal commands on the status bar when you enter them on the **TCC** command line or in the [Command Input](#) window. If you move the mouse over the syntax message on the status bar, **Take Command** will display a tooltip with the full syntax and switch descriptions.

4 TCC

TCC is a command processor compatible with CMD (the default command processor in Windows) but massively enhanced with thousands of additional features. **TCC/LE** is a limited version of **TCC**, distributed as a separate free product.

- [Comparing TCC and TCC/LE](#)
- [Starting TCC](#)

- › [Commands](#)
- › [Variables & Functions](#)
- › [The TCC Command Line](#)
- › [Aliases & Batch Files](#)
- › [File Selection](#)
- › [Input / Output Redirection](#)
- › [Configuration Options](#)
- › [IDE / Batch Debugger](#)

4.1 Comparing TCC, TCC/LE, and CMD

TCC comes in two versions: the full **TCC** as distributed with **Take Command**, and **TCC/LE**, which is distributed separately in a free but unsupported version. We will refer to **TCC** in this section to mean both **TCC** and **TCC/LE**, except where a feature is flagged **Not in LE**.

TCC/LE contains a subset of the **TCC** commands, variables, and variable functions. In addition, **TCC/LE** does not include the following **TCC** features:

- Batch IDE (debugger and editor)
- FTP, FTPS, SFTP, HTTP, HTTPS, and TFTP file access
- OpenAFS
- Internal Perl, Python, REXX, Ruby, and Tcl/tk support
- Active Scripting
- Command dialogs

TCC/LE does include the full command line editing, alias, filename completion, directory navigation, file selection, and regular expression support in **TCC**.

If an internal command, variable function, or internal variable is not supported in **TCC/LE**, the heading for that topic in this help file will include **Not in LE**.

- [Internal Commands](#)
- [Variables](#)
- [Variable Functions](#)

The following table lists the **TCC** commands supported in **TCC/LE** -- and, by way of comparison, with the default Windows command processor **CMD**. Most of the **CMD** commands have only a limited subset of the options and/or functionality in the equivalent **TCC** command. (Commands marked with a * in the **CMD** column are external Windows commands.)

TCC	TCC/LE	CMD
?	Y	
ACTIVATE	Y	
ALIAS	Y	
ASSOC	Y	Y
ASSOCIATE		
ATTRIB	Y	*
BATCOMP		
BDEBUGGER		
BEEP	Y	
BREAK	Y	Y
BREAKPOINT		
CALL	Y	Y

CANCEL	Y	
CD / CHDIR	Y	Y
CDD	Y	
CHCP	Y	*
CLIPMONITOR		
CLS	Y	Y
COLOR	Y	Y
COPY	Y	Y
DATE	Y	Y
DATEMONITOR		
DEBUGMONITOR		
DEBUGSTRING		
DEFER		
DEL / ERASE	Y	Y
DELAY	Y	
DESCRIBE	Y	
DESKTOP		
DETACH		
DIR	Y	Y
DIRHISTORY	Y	
DIRS	Y	
DISKMONITOR		
DO	Y	
DRAWBOX	Y	
DRAWHLINE	Y	
DRAWVLINE	Y	
ECHO	Y	Y
ECHOERR	Y	
ECHOS	Y	
ECHOERR	Y	
ECHOX		
ECHOXERR		
EJECTMEDIA	Y	
ENDLOCAL	Y	Y
ESET	Y	
EVENTLOG		
EVENTMONITOR		
EVERYTHING		
EXCEPT	Y	
EXIT	Y	Y
FFIND	Y	
FIREWIREMONITOR		
FOLDERMONITOR		
FONT		
FOR	Y	Y
FREE	Y	
FTYPE	Y	Y
FUNCTION		
GLOBAL	Y	
GOSUB	Y	
GOTO	Y	Y
GZIP		
HEAD		

HELP	Y	*
HISTORY	Y	
IDE		
IF	Y	Y
IFF	Y	
IFTP		
INKEY	Y	
INPUT	Y	
JABBER		
KEYBD	Y	
KEYS	Y	Y
KEYSTACK	Y	
LIST	Y	
LOADBTM		
LOADMEDIA	Y	
LOG	Y	
LUA		
MD / MKDIR	Y	Y
MEMORY	Y	
MKLINK	Y	Y
MKLNK	Y	
MOVE	Y	Y
MSGBOX	Y	
NETMONITOR		
ON	Y	
OPTION	Y	
OSD		
PATH	Y	Y
PAUSE	Y	Y
PDIR	Y	
PLAYAVI		
PLAYSOUND		
PLUGIN	Y	
POPD	Y	Y
POSTMSG		
PRINT		
PRIORITY		
PROCESSMONITOR		
PROMPT	Y	Y
PUSHD	Y	Y
QUERYBOX	Y	
QUIT	Y	
RD / RMDIR	Y	Y
REBOOT	Y	
RECYCLE		
REM	Y	Y
REN / RENAME	Y	Y
RESOLUTION		
RETURN	Y	
RExec		
RSHELL		
SCREEN	Y	
SCREENMONITOR		

SCRIPT		
SCRPUT	Y	
SELECT	Y	
SENDHTML		
SENDMAIL		
SERVICEMONITOR		
SERVICES		
SET	Y	Y
SETARRAY		
SETDOS	Y	
SETERROR		
SETLOCAL	Y	Y
SHIFT	Y	Y
SHORTCUT		
SHRALIAS	Y	
SMPP		
SNMP		
SNPP		
START	Y	Y
STATUSBAR		
SWITCH	Y	
SYNC		
TAIL		
TAR		
TASKBAR		
TASKDIALOG		
TASKEND		
TASKLIST		
TCDIALOG		
TCFILTER	Y	
TCTOOLBAR	Y	
TEE	Y	
TEXT	Y	
TIME	Y	Y
TIMER	Y	
TITLE	Y	Y
TOUCH	Y	
TPIPE		
TRANSIENT		
TREE	Y	*
TRUENAME	Y	
TYPE	Y	Y
UNALIAS	Y	
UNFUNCTION		
UNGZIP		
UNSET	Y	
UNSETARRAY		
UNTAR		
UNZIP		
USBMONITOR		
VBEEP		
VER	Y	Y
VERIFY	Y	Y

VIEW		
VOL	Y	Y
VSCRPUT	Y	
WAKEONLAN		
WEBFORM		
WEBUPLOAD		
WHICH	Y	
WINDOW	Y	
WMIQUERY		
Y	Y	
ZIP		
ZIPSFX		
7UNZIP		
7ZIP		

The following table lists the **TCC** internal variables supported in **TCC/LE** (none are included in CMD):

TCC	TCC/LE
<u>+</u>	Y
=	Y
!	Y
?	Y
<u>4ver</u>	Y
<u>?</u>	Y
<u>acstatus</u>	
<u>admin</u>	
<u>afswcell</u>	
<u>alt</u>	Y
<u>ansi</u>	Y
<u>batch</u>	Y
<u>batchline</u>	Y
<u>batchname</u>	Y
<u>batchtype</u>	
<u>battery</u>	
<u>batterylife</u>	
<u>batterypercent</u>	
<u>bdebugger</u>	
<u>bg</u>	Y
<u>boot</u>	Y
<u>build</u>	Y
<u>capslock</u>	Y
<u>cdroms</u>	
<u>childpid</u>	
<u>ci</u>	Y
<u>cmdline</u>	Y
<u>cmdproc</u>	Y
<u>cmdspec</u>	Y
<u>co</u>	Y
<u>codepage</u>	Y
<u>column</u>	Y

columns	Y
consoleb	
consolepids	
country	Y
cpu	
cpuusage	
ctrl	Y
cwd	Y
cwds	Y
cwp	Y
cwps	Y
_date	Y
_datetime	Y
_day	Y
detachpid	
disk	Y
dname	Y
dos	Y
dosver	Y
dow	Y
dowf	Y
dowi	Y
doy	Y
drives	Y
dst	
dvds	
echo	Y
editmode	
elevated	
errorlevel	Y
execarray	
execstr	Y
exit	Y
expansion	Y
fg	Y
ftperror	
hdrives	
hlogfile	Y
host	
hour	Y
hwprofile	
ide	
idleticks	
idow	Y
idowf	Y
iftp	
iftps	
imonth	Y
imonthf	Y
ininame	Y
ip	

ipadapter	
ipadapters	
iparpproxy	
ipdns	
ipdnsserver	
iprouting	
isodate	Y
isodowi	
isowdate	
isoweek	
isowyear	
_kbhit	Y
_lalt	Y
_lastdir	
_lastdisk	Y
_lctrl	Y
_logfile	Y
_lshift	Y
minute	Y
monitors	
month	Y
monthf	Y
_numlock	Y
openafs	
osbuild	Y
_parent	Y
_pid	
_pipe	Y
_ppid	
ralt	Y
_rctrl	Y
ready	
registered	
row	Y
rows	Y
rshift	Y
_rubrtype	
_rubyvalue	
scrolllock	Y
second	Y
selected	
service	
shell	Y
shift	Y
shortcut	
shralias	Y
startpath	Y
startpid	
stdin	
stdout	
stderr	

stzn	
stzo	
syserr	Y
tccver	Y
tcfilter	Y
tcfolder	Y
tclistview	
tctab	Y
tctabactive	
tctabs	
time	Y
transient	Y
tzn	
tzo	
unicode	
utctime	
utcdate	
utcdatetime	
utchour	
utcisodate	
utcminute	
utcsecond	
vermajor	
verminor	
version	
virtualbox	
virtualpc	
vmware	
vxpixels	
ypixels	
windir	Y
winfgwindow	Y
winname	Y
winsysdir	Y
winticks	
wintitle	Y
winuser	Y
winver	Y
wow64	
wow64dir	
x64	
xmouse	
xpixels	Y
xwindow	
year	Y
ymouse	
ypixels	Y
ywindow	

The following table lists the **TCC** variable functions supported in **TCC/LE** (none are included in CMD):

TCC	TCC/LE
@ABS	Y
@AFSCELL	
@AFSMOUNT	
@AFSPATH	
@AFSSYMLINK	
@AFSVOLID	
@AFSVOLNAME	
@AGEDATE	
@ALIAS	
@ALTNAMESPACE	Y
@ARRAYINFO	
@ASCII	Y
@ASSOC	
@ATTRIB	Y
@AVERAGE	
@ALLOC	
@BFREE	
@BPEEK	
@BPEEKSTR	
@BPOKE	
@BPOKESTR	
@BREAD	
@BWRITE	
@CAPI	
@CAPS	
@CDROM	Y
@CEILING	
@CHAR	Y
@CLIP	Y
@CLIPW	Y
@COLOR	
@COMMA	Y
@COMPARE	
@CONSOLE	
@CONVERT	
@CONSOLEB	
@COUNT	Y
@CRC32	Y
@CWD	Y
@CWDS	Y
@DATE	Y
@DATECONV	
@DAY	Y
@DEC	Y
@DECIMAL	Y
@DESCRIPT	Y
@DEVICE	Y

@DIGITS	Y
@DIRSTACK	
@DISKFREE	Y
@DISKTOTAL	Y
@DISKUSED	Y
@DOMAIN	
@DOW	Y
@DOWF	Y
@DOWI	Y
@DOY	Y
@DRIVETYPE	Y
@DRIVETYPEEX	
@ENUMSERVERS	
@ENUMSHARES	
@ERRTEXT	
@EVAL	Y
@EXEC	Y
@EXECARRAY	
@EXECSTR	Y
@EXETYPE	
@EXPAND	Y
@EXT	Y
@FIELD	Y
@FIELDS	Y
@FILEAGE	
@FILEARRAY	
@FILECLOSE	Y
@FILEDATE	Y
@FILEHANDLE	
@FILENAME	Y
@FILEOPEN	Y
@FILEREAD	Y
@FILEREADB	Y
@FILES	Y
@FILESEEK	Y
@FILESEEKL	Y
@FILESIZE	Y
@FILETIME	Y
@FILEWRITE	Y
@FILEWRITEB	Y
@FILTER	
@FINDCLOSE	Y
@FINDFIRST	Y
@FINDNEXT	Y
@FLOOR	
@FOLDERS	
@FONT	
@FORMAT	Y
@FORMATN	Y
@FORMATNC	Y
@FSTYPE	Y
@FTYPE	

@FULL	Y
@FUNCTION	
@GETDATE	
@GETDATETIME	
@GETDIR	Y
@GETFILE	Y
@GETFOLDER	Y
@GROUP	
@HISTORY	
@IDOW	Y
@IDOWF	Y
@IF	Y
@INC	Y
@INDEX	Y
@INIREAD	Y
@INIWRITE	Y
@INODE	
@INSERT	Y
@INSTR	Y
@INT	Y
@IPADDRESS	
@IPADDRESSN	
@IPALIASES	
@IPDESC	
@IPDHCP	
@IPGATEWAY	
@IPIPV6N	
@IPNAME	
@IPNAMEN	
@IPPHYSICAL	
@IPPORT	
@IPSERVICEALIAS	
ES	
@IPSUBNET	
@IPTYPE	
@IPWINS	
@IPWINSSERVER	
@IPZONEID	
@ISALNUM	Y
@ISALPHA	Y
@ISASCII	Y
@ISCNTRL	Y
@ISDIGIT	Y
@ISFLOAT	
@ISLOWER	
@ISODOWI	
@ISOWEEK	
@ISOWYEAR	
@ISPRIME	
@ISPRINT	Y
@ISPROC	
@ISPUNCT	Y

@ISSPACE	Y
@ISUPPER	
@ISXDIGIT	Y
@JUNCTION	
@LABEL	Y
@LCS	
@LEFT	Y
@LEN	Y
@LFN	Y
@LINE	Y
@LINES	Y
@LINKS	
@LOWER	Y
@LTRIM	Y
@LUA	
@MAKEAGE	
@MAKEDATE	
@MAKETIME	
@MAX	
@MD5	Y
@MIN	
@MONTH	Y
@MX	
@NAME	Y
@NUMERIC	Y
@OPTION	Y
@OWNER	
@PATH	Y
@PERL	
@PING	
@PID	
@PIDCOMMAND	
@PLUGINVER	
@PPID	
@PRIME	
@PRIORITY	
@PROCESSTIME	
@PYTHON	
@QUOTE	
@RANDOM	Y
@READSCR	Y
@READY	Y
@REGBREAD	
@REGBWRITE	
@REGCREATE	Y
@REGCOPYKEY	
@REGDELKEY	Y
@REGEX	
@REGEXINDEX	
@REGEXIST	Y

@REGEXSUB	
@REGQUERY	Y
@REGSET	Y
@REGSETENV	Y
@REGTYPE	Y
@REMOTE	Y
@REMOVABLE	Y
@REPEAT	Y
@REPLACE	Y
@REVERSE	Y
@REXX	
@RIGHT	Y
@RTRIM	Y
@RUBY	
@SCRIPT	
@SEARCH	
@SELECT	Y
@SERIAL	Y
@SERVICE	
@SFN	Y
@SHA1	
@SHA256	
@SHA384	
@SHA512	
@SHFOLDER	
@SIMILAR	
@SNAPSHOT	
@STRIP	Y
@SUBST	Y
@SUBSTR	Y
@SUMMARY	
@SYMLINK	
@SYSTEMTIME	
@TALNUM	
@TALPHA	
@TARCFILE	
@TARCOUNT	
@TARDFILE	
@TARFILEDATE	
@TARFILESIZE	
@TASCII	
@TCL	
@TCNTRL	
@TDIGIT	
@TIME	
@TIMER	Y
@TLOWER	
@TPRINT	
@TPUNCT	
@TRIM	Y
@TRUENAME	Y
@TRUNCATE	

@TSPACE	
@TUPPER	
@TXDIGIT	
@UNC	
@UNICODE	Y
@UNIQUE	Y
@UNQUOTE	Y
@UNQUOTES	Y
@UPPER	Y
@UUID	
@VERINFO	
@VERSION	
@WATTRIB	Y
@WILD	Y
@WINAPI	
@WINCLASS	
@WINCLIENTSIZE	
@WINEXENAME	
@WININFO	Y
@WINMEMORY	Y
@WINMETRICS	
@WINPID	
@WINPOS	
@WINSIZE	
@WINSTATE	
@WINSYSTEM	
@WINTITLE	
@WMI	
@WORD	Y
@WORDS	Y
@WORKGROUP	
@XMLCLOSE	
@XMLNODES	
@XMLOPEN	
@XMLXPATH	
@YEAR	Y
@ZIPCFILE	
@ZIPCFILESIZE	
@ZIPCOMMENT	
@ZIPCOUNT	
@ZIPDFILE	
@ZIPDFILESIZE	
@ZIPFILECOMMENT	
@ZIPFILEDATE	

TCC also includes 65+ [command variables](#) that are not in **TCC/LE** or CMD.

4.2 Starting TCC

You will typically start **TCC** in a **Take Command** tab window. But you can also start **TCC** (and **TCC/LE**) from a Windows shortcut, located:

- on the desktop, or
- in the Quick Launch bar, or
- in the **Programs** section of the **Start** menu (including its **Startup** subdirectory).

You may also start it from the **Start / Run** dialog.

The **TCC/LE** installer will optionally create both a **TCC/LE** folder or group (in the **Programs** section of the **Start** menu) and a desktop object (shortcut) which starts **TCC/LE**. Usually these are sufficient, but if you prefer, you can create multiple desktop objects or items to start **TCC/LE** with different startup commands or options, or to run different applications in the tab windows.

See [TCC Startup Options](#) for more information on startup command line options.

When you configure a **TCC** shortcut, place the full path and name for the file in the Command Line field, and put any startup options that you want passed to **TCC**. For example:

Command Line:	C:\Program Files\JPSoft\TCMD\TCC.EXE
Working directory:	C:\

You do not need to use the Change Icon button, because **TCC.EXE** already contains icons.

Each Windows program has a command line which can be used to pass information to the program when it starts. The command line is entered in the Command Line field for each shortcut or each item in a Program Manager group (or each item defined under another Windows shell), and consists of the name of the program to execute, followed by any startup options.

The **TCC** startup command line does not need to contain any information. When invoked with an empty command line, **TCC** will configure itself from the [TCMD.INI file](#), and then display a prompt and wait for you to type a command. However, you may add information to the [startup command line](#) that will affect the way **TCC** operates.

4.2.1 TCC Startup Options

The command line that starts **TCC** will typically include the program name with drive and path, followed by any options. For example:

```
"c:\program files\jpsoft\tcmd13\tcc.exe" @c:\jpsoft\tcmd.ini
```

Although the startup command line is usually very simple, you can add several options. You can do this manually in the Windows **RUN** dialog, in a Windows shortcut file (.LNK), at the **TCC** prompt or in a batch file (with or without using the internal [START](#) command). Each of these methods will start a new instance of the selected command processor, which will run in a new window, except when **TCC** is started from **TCC** (either at the command prompt or within a batch file) without the [START](#) command.

When you use a [pipe](#) in a command, either at the command prompt or in a batch file, **TCC** starts another instance of itself, using the same command line parameters (except as required for the pipe).

The complete syntax for the **TCC** startup command line is (all on one line):

```
d:\path\tcc.exe [d:\path][[/@d:\path\inifile][//directive=value...]] [/A /H /
I[IPSX]/L: /LA /LD /LF /LH /N/Q /S /T:bf /U /V /X ][/C |/K][command]
```

Do not include the square brackets shown in the command line above. They are there to indicate that the items within the brackets are optional. Some options are available only in specific products; see below for details.

If you include any of the options below, you should use them in the order that they are described. If you do not do so, you may find that they do not operate properly.

The command line must start with the path and name of the executable program file (**TCC.EXE**):

```
d:\path\tcc.exe
```

The additional items below may be included on the command line:

```
d:\path
```

If included, this second copy *d:\path* of **TCC** path must be identical to *d:\path* in the command line segment above. It sets the drive and directory where the program is stored, called the **COMSPEC** path. This option is included for compatibility with other character mode command processors, but is not needed in normal use. **TCC** can find its own directory without a **COMSPEC** path.

```
@d:\path\inifile OR
/@d:\path\inifile
```

This option sets the path and name of the [.INI file](#). You don't need this option if

- 1) your [.INI file](#) is named *TCMD.INI*, and
- 2) it is in one of the following directories:
 - 2.1) the same directory as **TCC**
 - 2.2) the %localappdata% directory

This option is most useful if you want to start the program with a specific and unique [.INI file](#).

To start **TCC** without any [.INI file](#), you can use the /I or /II options, or create an empty file and specify it as your [.INI file](#).

To get around a Windows limitation that causes the displayed command line of a shortcut to be truncated when a parameter begins with @, you can use the alternative syntax

```
/@d:\path\inifile
```

TCC will skip the leading slash.

```
//directive=value
```

This option tells **TCC** to treat the text appearing between the // and the next space or tab as a directive. The directive should be in the same format as a line in the [.INI file](#), but may not contain spaces, tabs, or comments. This option may be repeated. It is a convenient way to place a few simple directives on the startup line without having to modify or create a new [.INI file](#).

Directives on the command line override any corresponding directive in the [.INI file](#).

- /A This option causes the output of internal commands to a pipe or redirected to a file to be in ASCII when **TCC** starts. This is the default value, and isn't necessary unless you want to override a [Unicode Output](#) configuration option.
- /D Disable execution of AutoRun commands from Registry. If /D is not specified when **TCC** starts, it will look for and execute the following registry variables:

HKEY_LOCAL_MACHINE\Software\Microsoft\Command Processor\AutoRun

and / or

HKEY_CURRENT_USER\Software\Microsoft\Command Processor\AutoRun

See also the [AutoRun](#) configuration option.

- /H Start **TCC** in a hidden window. The window will not appear on the task bar, or in the Alt-tab list of applications.
- /I Don't load the .INI file, execute TCSTART or TCEXIT, or load plugins. You can optionally specify individual arguments:
 - /II Don't load the .INI file.
 - /IP Don't load plugins.
 - /IS Don't execute TCSTART.
 - /IX Don't execute TCEXIT.

- /L: Forces the use of local lists as the default for aliases, functions, directory history and command history, overriding any configuration options. This method allows you to use global lists as the default, but start a specific session with local aliases, functions and histories. See the topics [ALIAS](#), [FUNCTION](#), and [Local and Global History Lists](#) for more details. Note the required trailing colon (:)!
You can optionally specify individual arguments:

- /LA Forces the use of local aliases.
 - /LD Forces the use of a local directory history.
 - /LF Forces the use of local functions (not supported in **TCC/LE**).
 - /LH Forces the use of a local command history list.
- /N If **TCC** was started as a service, use the /N option to prevent **TCC** from being closed on a Windows CTRL_LOGOFF_EVENT (not supported in **TCC/LE**).
 - /Q Don't display version / copyright message (registered copies only; not supported in **TCC/LE**).
 - /S This option tells **TCC** that you do not want it to set up a Ctrl-C / Ctrl-Break handler. It is included for compatibility with CMD.

Warning: It may cause the system to operate incorrectly if you use this option without

other software to handle Ctrl-C and Ctrl-Break. This option should be avoided by most users.

- /T:*bf*** This option sets the foreground and background colors in the **TCC** command window. Both **b** and **f** are hexadecimal digits. **b** specifies the background color and **f** specifies the foreground color. This option is included only for compatibility with CMD. See the CMD color codes in [Colors, Color Names & Codes](#).

In most cases you should set default colors with the corresponding [Output Colors](#) configuration option. If you use both, the /T switch overrides the configuration options.

- /U** This option causes the output of internal commands to a pipe or redirected to a file to be in Unicode when **TCC** starts. The command :

[OPTION](#) //UnicodeOutput=yes | no

may be used at any time to switch between Unicode and ASCII output.

- /U8** This option causes the output of internal commands to a pipe or redirected to a file to be in UTF-8 when **TCC** starts. The command :

[OPTION](#) //UTF8Output=yes | no

may be used at any time to switch between UTF-8 and ASCII output.

- /V** Tells **TCC** to handle the CMD syntax **!varname!** as a delayed expansion of **%varname%**. Since CMD, unlike **TCC**, doesn't support delayed expansion of variable references in the **%varname%** format, it introduced a special **!varname!** notation. Using **/V** simply tells **TCC** to handle that syntax as an alternative to **%varname%** or **%varname!** or **[varname]**.

- /X** This option forces **TCC** to alter the operation of the [MD \(MKDIR\)](#) command to automatically create all necessary intermediate directories when it creates a new subdirectory. Its effect is the same as adding a /S option to all [MD \(MKDIR\)](#) commands. This option is included for compatibility with CMD, where it also enables other options. However, in **TCC** those options are already enabled by default.

/C command or

/K command or

command

Only one of these options may be used to specify for **TCC** what it must do after startup, and what it should do after completing **command**. **Command** will be executed after the automatic **TCC** startup program [TCSTART](#), but before a prompt is displayed. **Command** may be any valid alias, internal or external command, or batch file, including parameters.

All other startup options must be placed before **command**, because **TCC** will treat characters after **command** as parameters for **command** and not as additional startup options.

If **command** is preceded by **/C**, **TCC** will execute **command** and then exit, returning to the parent program or the desktop without displaying a prompt.

The **/K** switch has no effect. Using it is the same as placing **command** (with neither **/C** nor **/K**) at the end of the startup command line. It is included only for compatibility with CMD.

Example 1

Assume that you execute the command line below:

```
c:\TCMD\TCC.exe c:\TCMD\start.btm
```

The events below will take place in the order shown:

- 1 **Windows** starts **c:\TCC\TCC.exe**
- 2 **TCC** initializes from
 - 1st choice: **c:\TCC\TCMD.INI**
 - 2nd choice: TCMD.INI in the %localappdata% directory.
- 3.1 If the initialization file was found, **and** it contains the directive
TCStartPath=c:\start
and one of the files


```
c:\start\tcstart.btm
c:\start\tcstart.bat
c:\start\tcstart.cmd
c:\start\tcstart.exe
c:\start\tcstart.com
```

 exists, that file is executed by **TCC**.
- 3.2 If no initialization file was found in Step 2, **or** the initialization file either does not contain the **TCStartPath** directive, or the value of the directive is **c:\TCC**, and a **TCSTART** program is found in directory **c:\TCC**, it is executed by **TCC**
- 4 **TCC** executes **c:\tcmd\start.btm** (or, if not found, it displays an error message).
- 5 **TCC** displays the command prompt, unless an [EXIT](#) command was executed in **c:\tcmd\start.btm**, terminating **TCC**.

Example 2

The command line below, when executed by **TCC**, CMD, the RUN dialog, or a shortcut, will start **TCC**, select local aliases, execute any **TCSTART** file you have created, execute the file **PROCESS.BTM**, and exit. No prompt will be displayed by this session:

```
c:\tcmd13\tcc.exe /la /c c:\tcmd13\process.btm
```

4.2.2 TCSTART and TCEXIT

TCC Startup Program

Each time **TCC** starts, it looks for a program named [TCSTART](#). TCSTART is normally a batch file (**.BAT**, **.BTM**, or **.CMD**), but it can be any executable file. If you specify a path in the [TCSTART / TCEXIT](#) configuration option, the file must be in the specified directory. If the configuration option is not used, the **TCSTART** program, if any, in the same directory as your command processor is executed. Use of **TCSTART** is optional, and **TCC** will not display an error message if it cannot find the program. If you do not want to use a startup program, set the [TCSTART / TCEXIT](#) path to a directory which does not have one, or leave it unspecified, and make sure that no matching executable file is in **TCC**'s directory.

TCSTART is a convenient place to change the color or content of the prompt for each session, [LOG](#) the start of a session, or execute other special startup or configuration commands. It is also one way to set [aliases](#), [functions](#), and [environment](#) variables. See the section below on Pipes etc. about changing

directories via **TCSTART**.

With the exception of some [initialization switches](#), the entire startup command line passed to **TCC** is available to **TCSTART** as [batch file parameters](#) (%1, %2, etc.). For example, to pause if any parameters are passed, you could include this command in **TCSTART**:

```
if %# GT 0 pause Starting %_cmdproc with parameters [%$]
```

You can disable **TCSTART** and/or **TCEXIT**

Pipes, Transient Sessions / Processes, and **TCSTART**

When you set up the **TCSTART** program, remember that it is executed every time the command processor starts, including when running a [pipe](#) or when a transient copy of **TCC** is started with the [/c startup option](#). For example, suppose you enter a command line like this, which uses a pipe:

```
[c:\data] myprog | sort > out.txt
```

Normally this command would create the output file C:\DATA\OUT.TXT. However, if your **TCSTART** program changes to a different directory, the output file will be written there, not in C:\DATA. This is because **TCC** starts a second copy (instance) of itself to run the commands on the right hand side of the pipe, and that new copy executes **TCSTART** before processing the commands from the pipe.

The same thing occurs if you use a transient session (one started with the **/C** option) to run an individual command, then exit. The session will execute in the directory set by **TCSTART**, not the directory in which it was originally started (e.g., by specifying a working directory in a shortcut). For example, suppose you set up a desktop object with a command line like this, which starts a transient session:

Command:	d:\tc\tcmd.exe /c list myfile.txt
Working Directory:	c:\data

Normally this shortcut would [LIST](#) the file C:\DATA\MYFILE.TXT. However, if **TCSTART** changes the default to a different directory, **TCC** will look for *MYFILE.TXT* there, not in C:\DATA.

Similarly, any changes to environment variables, aliases, or other settings in **TCSTART** will affect all copies of **TCC**, including those used for pipes and transient sessions.

You can work around these potential problems with the [IF](#) or [IFF](#) commands and the [PIPE](#) and [TRANSIENT](#) internal variables. For example, to skip all **TCSTART** processing when running in a pipe or in a transient session, you could use a command like this at the beginning of **TCSTART**:

```
if %_pipe != 0 .or. %_transient != 0 quit
```

TCC Termination Program

Whenever a **TCC** session ends, it looks for a program named [TCEXIT](#). **TCEXIT** is normally a batch file (.BAT, .BTM, or .CMD), but it can be any executable file. The location of this optional program is determined by the same rule as the location of the **TCSTART** program for the session, and is not necessary in most circumstances. However, it is a convenient place to put commands to save information from one session to another, such as a (command) history list before **TCC** exits, or to [LOG](#) the end of the session. You can use a termination program even if you have no startup program.

No parameters are passed to the termination program.

4.2.3 TCC Exit Codes

If you start **TCC** from another program (e.g. to run a batch file or internal command), it will return a numeric code to the other program when it exits. This code indicates whether or not the operation performed was successful, with **0** indicating success and a non-zero value indicating a failure or other numeric result.

TCC's exit code is normally the numeric exit code from the last internal or external command. However, for CMD compatibility reasons and to avoid conflicts with external commands, only some internal commands set the exit code; others leave it unchanged from the most recent external command.

You can also use the [EXIT n](#) command to explicitly set the exit code. This overrides the rules above, and sets the return code to the parameter of your [EXIT](#) command.

4.3 Commands

TCC gives you instant access to more than 180 internal commands. (By contrast, Microsoft's [CMD](#) has fewer than 40 internal commands.) The best way to learn about commands is to experiment with them. This section will help you find the one(s) that you need, categorized in the lists below by name and by category.

- [Commands By Name](#)
- [Commands By Category](#)

Note: Remember that you can replace any internal command with an [ALIAS](#) or [plugin](#), or disable an internal command with [SETDOS /I](#).

4.3.1 Commands by Name

See also: [Internal Commands Listed by Category](#)

	Description
?	Display list of internal commands or Prompt to execute a command
ACTIVATE	Activate or set window state
ALIAS	Define or display aliases
ASSOC	Windows file associations
ASSOCIATE	Combine ASSOC and FTYPE
ATTRIB	Change or display file attributes
BATCOMP	Batch file compression
BDEBUGGER	Batch file debugger
BEEP	Beep the speaker
BREAK	Define or display Ctrl-C state
BREAKPOINT	Set a batch debugger breakpoint
CALL	Call another batch file
CANCEL	End batch file processing
CD	Display or change directory
CDD	Change drive and directory
CHCP	Display or change code page

CHDIR	Display or change directory
CLIPMONITOR	Monitor Windows clipboard
CLS	Clear the display window
COLOR	Change the display colors
COPY	Copy files and/or directories
DATE	Display or change date
DATEMONITOR	Monitor the current date and time
DEBUGMONITOR	Monitor OutputDebugString calls
DEBUGSTRING	Send text to system debugger
DEFER	Defer a command until batch file exit
DEL	Delete files and/or directories
DELAY	Wait for specified time
DESCRIBE	Display or change descriptions
DESKTOP	Create or switch desktops
DETACH	Start app detached
DIR	Display files and/or directories
DIRHISTORY	Display directory history list
DIRS	Display directory stack
DISKMONITOR	Monitor disk usage
DO	Create batch file loops
DRAWBOX	Draw a box
DRAWHLINE	Draw a horizontal line
DRAWVLINE	Draw a vertical line
ECHO	Echo a message
ECHOERR	Echo a message to STDERR
ECHOS	Echo a message with no CR/LF
Echoserr	Echo with no CR/LF to STDERR
ECHOX	Echo with no expansion to STDOUT
ECHOXERR	Echo with no expansion to STDERR
EJECTMEDIA	Eject a removable drive
ENDLOCAL	Restore from a SETLOCAL
ERASE	Delete files and/or directories
ESET	Edit variables or aliases
EVENTLOG	Write Windows event log
EVENTMONITOR	Monitor event log
EVERYTHING	Search for files and/or directories
EXCEPT	Exclude files from a command
EXIT	Exit TCC
FFIND	Search for files or text
FIREWIREMONITOR	Monitor FireWire devices
FOLDERMONITOR	Monitor folders and/or files
FONT	Change console font
FOR	Repeat a command
FREE	Display disk space
FTYPE	Display or edit file types

FUNCTION	Create or edit user functions
GLOBAL	Run command in subdirectories
GOSUB	Call batch subroutines
GOTO	Branch in a batch file
GZIP	Compress files to .gz archive
HEAD	Display beginning of file
HELP	Help for internal commands
HISTORY	Display or change history
IF	Conditional command execution
IFF	Conditional command execution
IFTP	Open FTP connection
INKEY	Get a single keystroke
INPUT	Get a text string
JABBER	Send an IM
KEYBD	Set keyboard toggles
KEYS	Enable or disable history list
KEYSTACK	Send keystrokes to app
LIST	Display content of files
LOADBTM	Load batch file as .BTM
LOADMEDIA	Close CD-ROM / DVD drive door
LOG	Save log of commands
LUA	Call the internal Lua interpreter
MD	Create subdirectories
MEMORY	Display memory statistics
MKDIR	Create subdirectories
MKLINK	Create NTFS symbolic links
MKLNK	Create NTFS hard or soft link
MOVE	Move files or directories
MSGBOX	Popup message box
NETMONITOR	Monitor networks
ON	Batch file error trapping
OPTION	Configure the TCC console
OSD	Display floating text
PATH	Set or display PATH
PAUSE	Wait for input
PDIR	User-formatted DIR
PLAYAVI	Display an .AVI file
PLAYSOUND	Play a sound file
PLUGIN	Load or unload plugin DLL
POPD	Restore from directory stack
POSTMSG	Send a message to a Window
PRINT	Print a file
PRIORITY	Set process priority
PROCESSMONITOR	Monitor processes
PROMPT	Change command line prompt

PUSHD	Save directory to stack
QUERYBOX	Popup input box
QUIT	Exit batch file
RD	Remove subdirectory
REBOOT	Reboot system
RECYCLE	Display or empty recycle bin
RESOLUTION	Change display resolution
RExec	Remotely execute commands
REM	Remark
REN	Rename files or directories
RENAME	Rename files or directories
RETURN	Return from GOSUB
RMDIR	Remove subdirectory
RSHELL	Remotely execute commands
SCREEN	Position cursor
SCREENMONITOR	Monitor screen saver
SCRPUT	Write directly to screen
SELECT	Select files for a command
SENDHTML	Send HTML email
SENDMAIL	Send email
SERVICEMONITOR	Monitor Windows services
SERVICES	Display, stop, or start system services
SET	Set or display environment variables
SETDOS	Set or display console options
SETLOCAL	Save environment, aliases & functions
SHIFT	Shift batch file parameters
SHORTCUT	Create a Windows shortcut
SHRALIAS	Share aliases
SMPP	Simple message transfer
SNMP	Send SNMP traps
SNPP	Send message to pager
START	Start a new session
STATUSBAR	Display text on status bar
SWITCH	Batch file switch / case
SYNC	Synchronize directories
TAIL	Display end of file
TAR	Add files to .tar archive
TASKBAR	Call Windows Taskbar functions
TASKDIALOG	Popup Windows task dialog
TASKEND	End a task
TASKLIST	Display Windows task list
TCDIALOG	Display command dialogs
TCFILTER	Filter Take Command List View
TCTOOLBAR	Edit Toolbar
TEE	Pipe "tee-fitting"

TEXT	Display text in batch file
TIME	Set or display time
TIMER	Stopwatch
TITLE	Set window title
TOUCH	Change file timestamps
TPIPE	Text filtering and substitution
TRANSIENT	Toggle shell transient mode
TREE	Display directory tree
TRUENAME	Display true pathname
TYPE	Display files
UNALIAS	Remove aliases
UNFUNCTION	Remove user-defined functions
UNSET	Remove environment variable
UNGZIP	Extract files from .gz archive
UNTAR	Extract files from .tar archive
UNZIP	Unzip files from zip archive
USBMONITOR	Monitor USB devices
VBEEP	Flash the screen and beep
VER	Display version
VERIFY	Display or set disk verification
VIEW	Display file contents
VOL	Display or set disk volume label
VSCRPUT	Write text vertically
WAKEONLAN	Send "Wake-On-LAN" packet
WEBFORM	Post data to web forms
WEBUPLOAD	Upload files to web servers
WHICH	Display command information
WINDOW	Window management
WMIQUERY	WMI queries
Y	Pipe "y-fitting"
ZIP	Zip files to zip archive
ZIPSFX	Create self-extracting executable
7UNZIP	Extract files from 7Zip archive
7ZIP	Compress files to 7Zip archive

4.3.2 Commands by Category

See also: [Internal Commands Listed by Name](#)

The best way to learn about commands is to experiment with them. The lists below categorize the available commands by topic and will help you find the one(s) you need.

- [File and directory management](#)
- [Subdirectory management](#)
- [Input and output](#)
- [Window management commands](#)
- [Commands primarily for use in or with batch files and aliases](#)

- ▶ [Environment and path commands](#)
- ▶ [System configuration and status](#)
- ▶ [Monitoring commands](#)
- ▶ [Compression / Decompression](#)
- ▶ [Other commands](#)

File and directory management

	Description
ATTRIB	Change or display file attributes
COPY	Copy files and/or directories
DEL	Delete files and/or directories
DESCRIBE	Display or change descriptions
ERASE	Delete files and/or directories
EVERYTHING	Search for files and/or directories
FFIND	Search for files or text
HEAD	Display beginning of file
IFTP	Open FTP connection
LIST	Display contents of files
MOVE	Move files or directories
RECYCLE	Display or empty recycle bin
REN	Rename files or directories
RENAME	Rename files or directories
SELECT	Select files for a command
SYNC	Synchronize directories
TAIL	Display end of file
TOUCH	Change file dates/times
TPIPE	Text filtering and substitution
TREE	Display directory tree
TRUENAME	Display true pathname
TYPE	Display files
UNZIP	Unzip files from archive
VIEW	Display file contents
Y	Pipe "y-fitting"
ZIP	Zip files to archive

Subdirectory management

	Description
CD	Display or change directory
CDD	Change drive and directory
CHDIR	Display or change directory
DIR	Display files and/or directories
DIRS	Display directory stack
MD	Create subdirectories
MKDIR	Create subdirectories
MKLNK	Create NTFS hard or soft link

PDIR	User-formatted DIR
POPD	Restore from directory stack
PUSHD	Save directory to stack
RD	Remove subdirectory
RMDIR	Remove subdirectory

Input and output

	<u>Description</u>
DRAWBOX	Draw a box
DRAWHLINE	Draw a horizontal line
DRAWVLINE	Draw a vertical line
ECHO	Echo a message
ECHOERR	Echo a message to stderr
ECHOS	Echo a message with no CR/LF
ECHOSERR	Echo with no CR/LF to stderr
ECHOX	Echo with no expansion
ECHOXERR	Echo with no expansion to stderr
FONT	Change console font
INKEY	Get a keystroke
INPUT	Get an input line
KEYSTACK	Send keystrokes to app
MSGBOX	Popup message box
OSD	Display floating text
PLAYAVI	Play an .AVI file
PLAYSOUND	Play a sound file
PRINT	Print a file
QUERYBOX	Popup input box
SCREEN	Position cursor
SCRPUT	Write directly to screen
SENDHTML	Send HTML email
SENDMAIL	Send email
SMPP	Send SMS message
SNMP	Send SNMP trap
SNPP	Send message to pager
STATUSBAR	Display text on status bar
TASKDIALOG	Popup Windows task dialog
VSCRPUT	Write text vertically

Window management commands

	<u>Description</u>
ACTIVATE	Activate or set window state
DESKTOP	Create or switch desktops
POSTMSG	Send a message to a Window
TITLE	Set window title

WINDOW	Window management
---------------	-------------------

Commands primarily for use in or with batch files and aliases
 (some work only in batch files; see the individual commands for details)

	Description
ALIAS	Define or display aliases
BATCOMP	Batch file compression
BDEBUGGER	Batch file debugger
BEEP	Beep the speaker
BREAKPOINT	Set a batch debugger breakpoint
CALL	Call another batch file
CANCEL	End batch file processing
DEBUGSTRING	Send text to system debugger
DEFER	Defer a command until the batch file exits
DELAY	Wait for specified time
DO	Batch file looping
ENDLOCAL	Restore a SETLOCAL
EJECTMEDIA	Eject a removable drive
FOR	Repeat a command
FUNCTION	Create or edit user functions
GLOBAL	Run command in subdirectories
GOSUB	Call batch subroutines
GOTO	Go to a batch file label
IF	Conditional command execution
IFF	Conditional command execution
JABBER	Send an IM
LOADBTM	Load batch files as .BTM
LOADMEDIA	Close CD-ROM / DVD drive door
ON	Batch file error trapping
PAUSE	Wait for input
QUIT	Exit batch file
REM	Remark
RETURN	Return from GOSUB
SETLOCAL	Save environment, aliases, and functions
SHIFT	Shift batch file parameters
SWITCH	Batch file switch / case
TEXT	Display text in batch file
TRANSIENT	Toggle shell transient mode
UNALIAS	Remove aliases
UNFUNCTION	Remove user-defined functions
VBEEP	Flash the screen and beep
WEBFORM	Post data to web servers
WEBUPLOAD	Upload files to web servers

Environment and path commands

	Description
ESET	Edit variables or aliases
PATH	Set or display PATH
SET	Set or display environment variables
UNSET	Remove environment variables

System configuration and status

	Description
ASSOC	Windows file associations
ASSOCIATE	Combine ASSOC and FTYPE
BREAK	Define or display Ctrl-C state
CHCP	Display or change code page
CLS	Clear the display window
COLOR	Change the display colors
DATE	Display or change date
DIRHISTORY	Display directory history list
EVENTLOG	Write to Windows event log
FREE	Display disk space
FTYPE	Display or edit file types
HISTORY	Display or change history
KEYBD	Set keyboard toggles
KEYS	Enable or disable history list
LOG	Save log of commands
MEMORY	Display memory statistics
OPTION	Configure the TCC console
PLUGIN	Load or unload plugin DLL
PROMPT	Change command line prompt
REBOOT	Reboot system
RESOLUTION	Change display resolution
SETDOS	Internal options
SERVICES	Display, stop, or start services
SHORTCUT	Create a Windows shortcut
TASKBAR	Call Windows Taskbar functions
TASKEND	End a task
TASKLIST	Display Windows task list
TCFILTER	Filter Take Command List View
TCTOOLBAR	Edit Take Command toolbar
TIME	Set or display time
VERIFY	Display or set disk verification
VER	Display version
VOL	Display or set disk volume label
WAKEONLAN	Send "Wake-On-LAN" packet

Monitoring commands

	Description
CLIPMONITOR	Monitor Windows clipboard
DATEMONITOR	Monitor current date and time
DEBUGMONITOR	Monitor OutputDebugString API
DISKMONITOR	Monitor disk usage
EVENTMONITOR	Monitor event log
FIREWIREMONITOR	Monitor FireWire devices
FOLDERMONITOR	Monitor folders and/or files
NETMONITOR	Monitor network connections
PROCESSMONITOR	Monitor processes
SCREENMONITOR	Monitor Windows screen saver
SERVICEMONITOR	Monitor Windows services
USBMONITOR	Monitor USB devices

Compression / Decompression commands

	Description
GZIP	Compress files to .gz archive
TAR	Add files to tar archive
UNGZIP	Extract files from .gz archive
UNTAR	Extract files from tar archive
UNZIP	Unzip files from archive
ZIP	Zip files to archive
ZIPSFX	Create self-extracting executable
7UNZIP	Extract files from 7Zip archive
7ZIP	Compress files to 7Zip archive

Other commands

	Description
?	Display list of internal commands, or prompt to execute a command
DETACH	Start app detached
EXCEPT	Exclude files from a command
EXIT	Exit TCC
HELP	TCC help
LUA	Call the internal Lua interpreter
SHRALIAS	Share aliases & functions
START	Start a new session
RExec	Remotely execute command
RSHELL	Remotely execute command
TCDIALOG	Display command dialogs
TEE	Pipe "tee-fitting"
TIMER	Stopwatch

WHICH	Display command information
-------	-----------------------------

4.3.3 Command Dialogs

Most of the internal TCC file handling commands have an alternate dialog form. This simplifies invoking the command when using some of the more obscure options, and also allows you to test commands and copy the generated command line to your batch files.

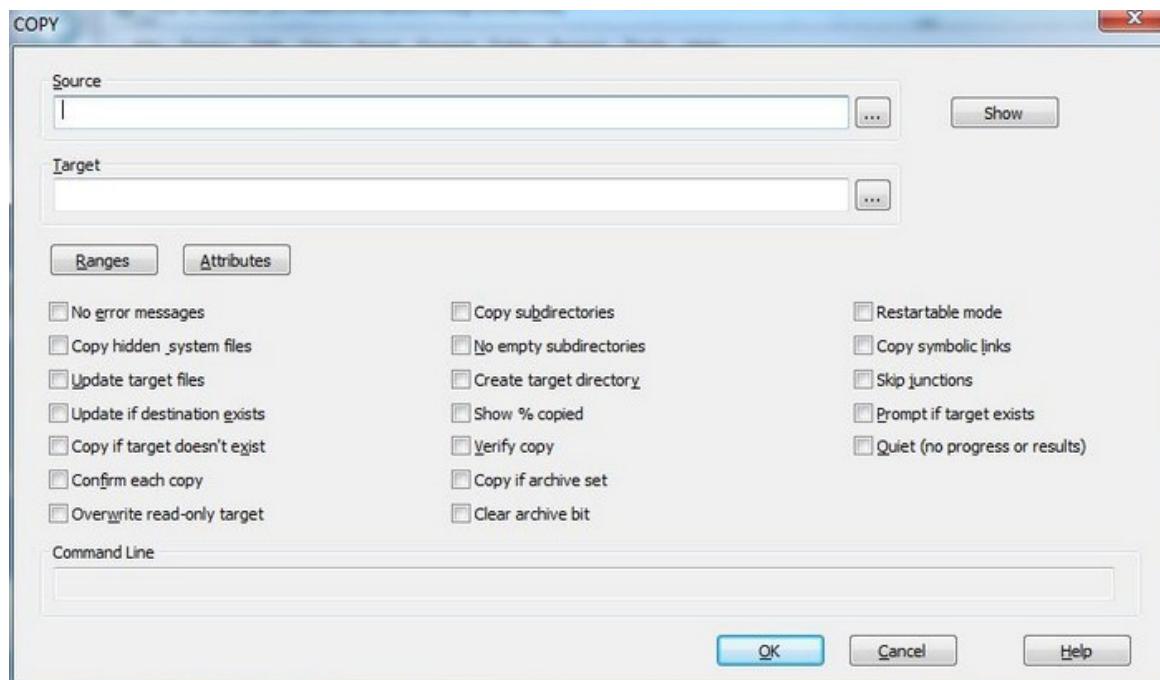
You can invoke the command dialogs three different ways:

- With the [TCDIALOG](#) command
- With the `/=` command line argument
- With the Alt-F1 key after entering the command name at the command line

The command dialog will be displayed, and when you press OK it will pass the generated command line on to the command.

For example:

```
tcdialog copy
```



The "Show" button in a command dialog will show all of the files that match the file specification in the edit field to the left. This may include subdirectories if you've selected that option (for example, in COPY or DIR), so it may take a few seconds to populate the list before displaying it.

4.3.4 ?

Purpose: Display a list of internal and plugin commands, or prompt for a command

Format: ? ["prompt" command]

Usage:

The ? command has two separate meanings:

1. When you use the ? command by itself, it displays a list of internal and plugin commands. For help with any individual command, see the [HELP](#) command. If you have disabled a command with [SETDOS](#) [/L](#), it will not appear in the list.
2. The second function of ? is to prompt the user before executing a specific command line. If you add a **prompt** and a **command**, ? will display the prompt followed by **(Y/N)?** and wait for the user's response. If the user presses **Y** or **y**, the command line will be executed. If the user presses **N** or **n**, it will be ignored.

Example

```
? "Load the network" call netstart.btm
```

When this command is executed, you will see the prompt

```
Load the network (Y/N) ?
```

If you answer **Y**, the [CALL](#) command will be executed:

4.3.5 ACTIVATE

Purpose: Activate a window, set its state, or change its title

Format: ACTIVATE [/R] "title" [MAX | MIN | RESTORE | DESKTOP CLOSE | ENABLE | DISABLE | TOPMOST | NOTOPMOST | TOP | BOTTOM | HIDE | /FLASH=type,count | /ICON=iconfile | /POS=left,top,width,height | /TRANS=n | TRAY | "newtitle"]

title	Current title of the window to be activated
left	New location of the left border of the window, in pixels
type	One or more of the following values: 0 - stop flashing 1 - flash the window caption 2 - flash the taskbar button 4 - flash continuously until WINDOW is called again with the /FLASH type set to 0 12 - flash continuously until the window comes to the foreground (cannot be used with 4)
count	Number of times to flash the window
top	New location of the top border of the window, in pixels
iconfile	New caption / task bar icon (an .ico file or an executable)
width	New width of the window, in pixels
height	New height of the window, in pixels
newtitle	New title for window
/R (estore original window)	

See also: [START](#), [TITLE](#), and [WINDOW](#).

Usage:

[ACTIVATE](#) activates, and optionally modifies, another session's window. It is not intended to modify the characteristics of the current **TCC** session (use [TITLE](#) or [WINDOW](#) for that purpose).

Title specifies the name of the target window to be activated. You can use [wildcards](#), including extended wildcards, in **title**. This is useful with applications that change their window title to reflect the file currently in use. **Title** must be enclosed in quotes.

If **title** begins with a =, it is assumed to be a process ID instead of a title. (Note that this is less reliable than providing a title, as a process can have multiple top-level windows.)

Each execution of ACTIVATE allows you to modify one property of the target window. To perform multiple operations, use multiple ACTIVATE commands.

The options are:

MAX	Expands the window to its maximum size and activates it.
MIN	Reduces the window to an icon.
RESTORE	Activates the window at its default size and location.
DESKTOP	Activates the Windows desktop.
CLOSE	Sends a "close" message to close the window.
ENABLE	Enable mouse and keyboard input.
DISABLE	Disable mouse and keyboard input.
TOPMOST	Keeps the window on top of all other windows until it closes, or NOTOPMOST is used.
NOTOPMOST	Allows other windows to overlay the window (this is the normal state for most windows).
TOP	Moves the window to the top of the window order, above all other non- TOPMOST windows.
BOTTOM	Moves the window to the bottom of the window order.
HIDE	Makes the window invisible (to make the window visible again, use RESTORE).
FLASH	Flash the window (not available in TCC/LE).
ICON	Change the window's caption and task bar icon.
POS	Sets the window position and size (in pixels).
TRANS	Transparency level, where n=0 (invisible) to 255 (opaque) (does not work for console windows).
TRAY	Move the specified window to the system tray.
"newtitle"	Changes the window title.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

If you specify **newtitle**, it must be enclosed in double quotes (which will not appear as part of the title text).

ACTIVATE is often used before [KEYSTACK](#) to make sure the proper window receives the keystrokes.

ACTIVATE works by sending messages to the named **window**. If the window ignores or misinterprets

the messages, ACTIVATE may not have the effect you want.

If ACTIVATE is used in a batch file, and the batch file is not itself running in the active window (the window with its title bar highlighted), then ACTIVATE may not activate the desired window. This is because under Windows you cannot make another window active except when the window which issues the command is itself active already. This is a Windows feature which helps to prevent windows which are not in the foreground from grabbing input intended for other windows.

Example

The example below first maximizes, and then renames the window originally called "Take Command":

```
activate "Take Command" max
activate "Take Command" "Take Command is Great!"
```

4.3.6 ALIAS

Purpose: Create new command names that execute one or more commands or redefine default options for existing commands; assign commands to keystrokes; load or display the list of defined alias names

Format: Display mode:
ALIAS [/G /L /P] [wildname]

Definition mode:
ALIAS [/G /L /O] [/R [/Z] file...] | name[=]value

file	One or more input files to read alias definitions from
wildname	Name of alias whose definition is to be displayed (may contain * and ? wildcards)
name	Name for an alias, or for the key to execute the alias
value	Text to be substituted for the alias name or key

<u>/G(lobal)</u>	<u>/P(ause)</u>
<u>/L(ocal)</u>	<u>/R(ead file)</u>
<u>/O(verwrite)</u>	<u>/Z (replace list)</u>

See also: [UNALIAS](#), [ESET](#), and [Aliases](#).

Usage:

- ▶ [Overview](#)
- ▶ [Displaying Aliases](#)
- ▶ [Multiple Commands and Special Characters in Aliases](#)
- ▶ [Nested Aliases](#)
- ▶ [Temporarily Disabling Aliases](#)
- ▶ [Partial \(Abbreviated\) Alias Names](#)
- ▶ [Keystroke Aliases](#)
- ▶ [Directory Aliases](#)
- ▶ [Saving and Reloading Your Aliases](#)
- ▶ [Alias Parameters](#)
- ▶ [Expanding Aliases at the Prompt](#)
- ▶ [Local and Global Aliases](#)

- › [Retaining Global Aliases with SHRALIAS](#)
- › [The PRE_INPUT, PRE_EXEC, and POST_EXEC Aliases](#)
- › [The UNKNOWN CMD Alias](#)
- › [Warnings](#)

Overview

The ALIAS command lets you create new command names or redefine internal commands. It also lets you assign one or more commands to a single keystroke. An alias is often used to execute a complex series of commands with a few keystrokes or to create "in memory batch files" that run much faster than disk-based batch files.

For example, to create a single-letter command **d** to display a wide directory, instead of using [DIR /W](#), you could use the command:

```
alias d = dir /w
```

Now when you type a single **d** as a command, it will be translated into a **DIR /W** command.

If an ALIAS command specifies a **value**, and there was an alias already assigned to **name**, the old alias value is discarded.

If you define aliases for commonly used application programs, you can often remove the directories they're stored in from the [PATH](#). For example, if you use Microsoft Word and had the **C:\WINWORD** directory in your path, you could define the following alias:

```
alias ww = c:\winword\winword.exe
```

With this alias defined, you can probably remove **C:\WINWORD** from your path. Word will now load more quickly than it would if **TCC** had to search the [PATH](#) for it. In addition, [PATH](#) can be shorter, which will speed up searches for other programs.

If you apply this technique for each application program, you can often reduce your [PATH](#) to just two or three directories containing utility programs, and significantly reduce the time it takes to load most software on your system. Before removing a directory from the [PATH](#), you will need to define aliases for all the executable programs you commonly use which are stored in that directory.

TCC also supports [Directory Aliases](#), a shorthand way of specifying pathnames.

Aliases are stored in memory, and are not saved automatically when you turn off your computer or end your current **TCC** session. See below for information on saving and reloading your aliases.

Displaying Aliases

If you want to see a list of all currently defined aliases, type:

```
alias
```

You can view the definition of a single alias. For example, if you want to see the definition of the alias **LIST**, you can type:

```
alias list
```

You can also view the definitions for all aliases matching a specific pattern by specifying a single parameter containing wildcards (* or ?). For example:

```
alias *win*
```

will display all aliases containing the string **win**.

You can use the [/P](#) option to control display scrolling when displaying aliases.

Multiple Commands and Special Characters in Aliases

An alias can represent more than one command. For example:

```
alias letters = `cd \letters & tedit`
```

This alias creates a new command called **LETTERS**. The command first uses [CD](#) to change to a subdirectory called **\LETTERS** of the directory current at the time of its execution, and then runs a program called **TEDIT**.

Aliases make extensive use of the [command separator](#) and the [parameter character](#), and may also use the [escape character](#).

When an alias contains multiple commands, the commands are executed one after the other. However, if any of the commands runs an external Windows application, you must be sure the alias will wait for the application to finish before continuing with the other commands. This behavior is controlled by the **Wait for completion** setting in the [configuration dialogs](#).

When you use the alias command at the command prompt or in a batch file, you must use back quotes ` around the alias definition if it contains multiple commands, or parameters (discussed below), or environment variables, or variable functions, or redirection, or piping. If you do not use back quotes, parameters, variables and functions are evaluated, and redirection or piping performed during the alias definition, and only the first command becomes part of the alias, the remaining ones are performed immediately. The back quotes prevent this premature expansion. You may use back quotes around other definitions, but they are not required. You do not need back quotes when your aliases are loaded from an ALIAS /R file; see below for details. The examples above and below include back quotes only when they are required.

Nested Aliases

Aliases may invoke internal commands, external commands, or other aliases. However, an alias may not invoke itself, except in special cases where an [IF](#) or [IFF](#) command is used to prevent an infinite loop. The two aliases below demonstrate alias nesting (one alias invoking another). The first line defines an alias which runs in the current directory, and executes **Word** located in the **E:\WINWORD**. The second alias changes directories with the [PUSHD](#) command, runs the **WP** alias, and then returns to the original directory with the [POPD](#) command:

```
alias wp = e:\winword\winword.exe
alias w = `pushd c:\wp & wp & popd`
```

The second alias above could have included the full path and name of **WINWORD.EXE** instead of calling the **WP** alias. However, writing two aliases makes the second one easier to read and understand, and makes the first alias available for independent use. If you rename the **WINWORD.EXE** program or move it to a new directory, only the first alias needs to be changed.

Temporarily Disabling Aliases

If you put an asterisk * immediately before a command in the **value** of an alias definition (the part after the equal sign), it tells **TCC** not to attempt to interpret that command as another (nested) alias. An asterisk used this way must be preceded by a space or the command separator and followed immediately by an internal or external command name.

By using an asterisk, you can redefine the default options for any internal or external command. For example, suppose that you always want to use the [DIR](#) command with the /2 (two column) and /P (pause at the end of each page) options:

```
alias dir = *dir /2/p
```

If you didn't include the asterisk, the second DIR on the line would be the name of the alias itself, and **TCC** would repeatedly re invoke the [DIR](#) alias, rather than running the [DIR](#) command. This would cause an "Alias loop" or "Command line too long" error. The asterisk forces interpretation of the second [DIR](#) as a command, not an alias.

An asterisk also helps you keep the names of internal commands from conflicting with the names of external programs. For example, suppose you have a program called *DESCRIBE.EXE*. Normally, the internal [DESCRIBE](#) command will run anytime you type DESCRIBE. But two simple aliases will give you access to both the *DESCRIBE.EXE* program and the [DESCRIBE](#) command:

```
alias describe = c:\winutil\describe.exe
alias filedesc = *describe
```

The first line above defines **describe** as an alias for the *DESCRIBE.EXE* program. If you stopped there, the external program would run every time you typed DESCRIBE and you would not have easy access to the internal [DESCRIBE](#) command. The second line defines **FILEDESC** as a new name for the internal [DESCRIBE](#) command. The asterisk is needed in the second command to indicate that the following word means the internal command [DESCRIBE](#), not the **describe** alias which runs your external program.

Another way to understand the asterisk is to remember that a command is always checked for an alias first, then for an internal or external command, or a batch file. The asterisk at the beginning of a command name simply skips over the usual check for aliases when processing that command, and allows **TCC** to go straight to checking for an internal command, external command, or batch file.

You can prevent alias expansion by using an asterisk before a command that you enter at the command line or in a batch file. This can be useful when you want to be sure you are running the original command and not an alias with the same name, or temporarily defeat the purpose of an alias which changes the meaning or behavior of a command. For example, above we defined an alias for [DIR](#) which made directories display in 2-column paged mode by default. If you wanted to see a directory display in the normal single-column, non-paged mode, you could enter the command ***DIR** and the alias would be ignored for that command.

You can disable aliases temporarily with the [SETDOS /X](#) command.

Partial (Abbreviated) Alias Names

You can also use an asterisk in the **name** of an alias. When you do, the characters following the asterisk are optional when you invoke the alias command. (Use of an asterisk in the alias **name** is unrelated to the use of an asterisk in the alias **value** discussed above.) For example, with this alias:

```
alias wher*eis = dir /s /p
```

The new command, **WHEREIS**, can be invoked as WHER, WHERE, WHEREI, or WHEREIS. Now if you type:

```
where myfile.txt
```

The **WHEREIS** alias will be expanded to the command:

```
dir /s /p myfile.txt
```

Keystroke Aliases

There are two kinds of keystroke aliases: [insert-only](#) and [autoexecute](#).

Insert-only Keystroke Aliases

Assignment: To assign an insert-only alias to a keystroke, use the key name on the left side of the equal sign, preceded by one at sign @, and the value of the alias on the right side of the equal sign:

```
alias @key=value
```

Operation: When you press the key to which you assigned an insert-only alias, **TCC** displays and inserts the alias value in the current command line, at the current cursor position. If your command line editing mode is overwrite, and the cursor is not at the end of the line, the alias value will overwrite part of the command line. You can continue to edit the command line, e.g., adding other parameters to the command. You must press **Enter** to execute the command.

Examples:

To assign the command **DIR /W** to the **F4** key, type:

```
alias @F4 = dir /w
```

To use it, press **F4** at the command prompt, and **DIR /W** will be placed on the command line for you. You can type additional parameters if you wish, and press **Enter** to execute the command. With the example alias, you can define the files that you want to display after pressing **F4** and before pressing **Enter** to execute the command.

You can also define a keystroke alias to insert a frequently used string into the middle of a command, e.g.,

```
alias @shift-F4 =%@expand[
```

which specific example can assist in processing wildcards for a program without such a feature.

Autoexecute Keystroke Aliases

Assignment: To assign an autoexecute alias to a keystroke, use the key name on the left side of the equal sign, preceded by two at signs @@, and the value of the alias on the right side of the equal sign:

```
alias @@key=value
```

Operation: When you press the key to which you assigned an autoexecute alias, **TCC** inserts the alias value in the current command line, at the current cursor position. If your command line editing mode is overwrite, and the cursor is not at the end of the line, the alias value will overwrite part of the command line. After the insertion/overwrite the command line is automatically executed.

Example: This command will assign an alias to the **F11** key that uses the [CDD](#) command to take you back to the previous default directory:

```
alias @@f11 = cdd -
```

Special Considerations for Keystroke Aliases

When you define keystroke aliases, the assignments will only be in effect at the command line, not inside application programs or batch files.

To insure that a keystroke alias, esp. an autoexecute one, is on the command line by itself, use the character defined by the [EraseLine](#) option (by default, the **Esc** key, represented as **^e** or **%=e**) as the first character of the alias value.

To force a visible indication that an autoexecute keystroke alias was used, include a descriptive [ECHO](#) command in the alias value.

Be careful not to assign aliases to keys that are already used at the command line (e.g., **F1** for [HELP](#)). The command line meanings take precedence and the keystroke alias will never be invoked. If you want to use one of the command line keys for an alias instead of its normal meaning, you must first disable its default use with the [NormalKey](#) or [NormalEditKey](#) options.

The **value** of an alias, including a keystroke alias, may contain only characters. It cannot contain representations of keys such as **F1 .. F12**, **Home**, etc.

See [Keys and Key Names](#) for a complete listing of key names and a description of the key name format.

Directory Aliases

Directory Aliases are a shorthand way of specifying pathnames. For example, if you define an alias:

```
alias pf:=c:\program files
```

You can then reference the files in **c:\program files\pjsoft** by entering **pf:\pjsoft**. Directory aliases work in places that accept filenames and directory names (internal command arguments or the first argument in a command line), including filename completion. You cannot use them in arguments to external applications, as **TCC** has no way of knowing what is a valid argument for external applications.

Directory alias names can be either two or more alphanumeric characters followed by a colon, or a single digit followed by a colon. You cannot [abbreviate](#) directory aliases.

Directory aliases support environment variable expansion.

Saving and Reloading Your Aliases

You can save your aliases to a file:

```
alias > alias.lst
```

You can then reload all the alias definitions in the file the next time you start up with the command:

```
alias /r alias.lst
```

This is much faster than defining each alias individually in a batch file. If you keep your alias definitions in a separate file which you load when **TCC** starts, you can edit them with a text editor, reload the edited file with **ALIAS /R**, and know that the same alias list will be loaded the next time you start **TCC**.

When you define aliases in a file that will be read with the **ALIAS /R** command, you do not need back quotes around the value, even if back quotes would normally be required when defining the same alias at the command line or in a batch file.

To remove an alias, use the [UNALIAS](#) command.

Alias Parameters

Aliases can use command line parameters or parameters like those in batch files. The command line parameters are numbered from %0 to %511. (%0 contains the alias name.) You can use double quotes to pass spaces, tabs, commas, and other special characters in an alias parameter; see [Parameter Quoting](#) for details. (Alias examples in this section assume the **TCC** default of ParameterChar=\$.)

Parameters that are referred to in an alias, but which are missing on the command line, appear as empty strings inside the alias. For example, if you only put two parameters on the command line, any reference in the alias to %3 or any higher-numbered parameter will be interpreted as an empty string.

The parameter %n\$ has a special meaning. **TCC** interprets it to mean "the entire command line, from parameter *n* to the end." If *n* is not specified, it has a default value of 1, so %\$ means "the entire command line after the alias name."

The parameter %-n\$ means "the command line from parameter 1 to *n* - 1".

The special parameter %# contains the number of command line parameters.

For example, the following alias will change directories, perform a command, and return to the original directory:

```
alias in `pushd %1 & %2$ & popd`
```

When this alias is invoked as:

```
in c:\comm mycomm /zmodem /56K
```

The first parameter, %1, has the value **c:\comm**. %2 is **mycomm**, %3 is **/zmodem**, and %4 is **/56K**. The command line expands into these three separate commands:

```
pushd c:\comm  
mycomm /zmodem /56K  
popd
```

This next example uses the [IFF](#) command to redefine the defaults for [SET](#). It should be entered on one line:

```
alias set = `iff %# == 0 then & *set /p & else & *set %$ &endiff`
```

This modifies the [SET](#) command so that if [SET](#) is entered with no parameters, it is replaced by SET /P (pause after displaying each page), but if [SET](#) is followed by a parameter, it behaves normally. Note the use of asterisks (*set) to prevent alias loops.

If an alias uses parameters, command line parameters will be deleted up to and including the highest referenced parameter. For example, if an alias refers only to %1 and %4, then the first and fourth parameters will be used, the second and third parameters will be discarded, and any additional parameters beyond the fourth will be appended to the expanded command (after the **value** portion of the alias). If an alias uses no parameters, all of the command line parameters will be appended to the expanded command. A convenient way to prevent unwanted command line parameters from being appended is to add a reference to %511 within the alias.

Aliases also have full access to all variables in the environment, internal variables, and variable functions. For example, you can create a simple command line calculator this way:

```
alias calc = `echo The answer is: %@eval[%$]`
```

Now, if you enter:

```
calc 5 * 6
```

The alias will display:

```
The answer is: 30
```

Expanding Aliases at the Prompt

You can expand an alias on the command line and view or edit the results by pressing **Ctrl-F** after typing the alias name, but before the command is executed. This replaces the alias with its contents, and substitutes values for each alias parameter, just as if you had pressed the **Enter** key. However, the command is not executed; it is simply redisplayed on the command line for additional editing.

Ctrl-F is especially useful when you are developing and debugging a complex alias, or if you want to make sure that an alias that you may have forgotten won't change the effect of your command.

Local and Global Aliases

Aliases can be stored in either a local or global list. The selection is made during **TCC** startup, using the **/L** or **/LA START** or [startup options](#), or by the [Local Aliases](#) configuration option, or interactively with the **ALIAS /G** and **ALIAS /L** options. The global alias list is limited to 256 K characters; the local alias list is limited only by memory size.

With a local alias list, any changes made to the aliases will only affect the current copy of **TCC**. They will not be visible in other shells or other sessions.

With a global alias list, all copies of **TCC**, which are started with global alias list will share the same alias list, and any changes made to the aliases in one copy will affect all other copies. This is the default for **TCC**.

There is no fixed rule for determining whether to use a local or global alias list. Depending on your work

style, you may find it most convenient to use one type, or a mixture of types in different sessions or shells. We recommend that you start with the default approach, then modify it if you find a situation where the default is not convenient.

When you use [SETLOCAL](#) / [ENDLOCAL](#) inside a batch file, changes in alias definitions are restored by the [ENDLOCAL](#). However, if the session uses the global alias list, any concurrent sessions also using the global alias list are affected.

Retaining Global Aliases with SHRALIAS

If you select a global alias list for **TCC** you can share the aliases among all running copies of **TCC**. When you close all **TCC** sessions, the memory for the global alias list is released, and a new, empty alias list is created the next time you start **TCC**.

If you want the alias list to be retained in memory even when no **TCC** session is running, you need to execute the [SHRALIAS](#) command, which performs this service for the global alias list, the global user-defined functions list, the global command history list, and the global directory history list. You may find it convenient to execute [SHRALIAS](#) from your [TCSTART](#) file.

[SHRALIAS](#) retains the alias list in memory, but cannot preserve it when Windows itself is shut down. To save your aliases when restarting Windows, you must store them in a file and reload them after the system restarts. For details on how to do so, see [Saving and Reloading Your Aliases](#) above.

The PRE_INPUT, PRE_EXEC, and POST_EXEC Aliases

When at the command prompt (i.e., not executing a batch file), **TCC** will look for (and execute them if found) the following aliases:

PRE_INPUT - executed immediately before accepting input for a new command line.

PRE_EXEC - executed immediately after a command line is entered (before any expansion or redirection).

POST_EXEC - executed immediately after returning from a command and before displaying the prompt.

None of these aliases will be passed any arguments.

If the alias does not exist, **TCC** will search the [plugins](#) for **PRE_INPUT** / **PRE_EXEC** / **POST_EXEC** functions and execute them if found.

The UNKNOWN_CMD Alias

If you create an alias with the name **UNKNOWN_CMD**, it will be executed any time **TCC** would normally issue the "Unknown command" error message. This allows you to define your own handler for unknown commands. When the **UNKNOWN_CMD** alias is executed, the command line which generated the error is passed to the alias for possible processing. For example, to just display the command that caused the error:

```
alias unknown_cmd `echo Error in command "%$"`
```

If the **UNKNOWN_CMD** alias contains an unknown command, it will call itself repeatedly. If this occurs,

TCC will loop up to 10 times, then display the **UNKNOWN_CMD loop** error.

If an **UNKNOWN_CMD** alias does not exist, **TCC** will search the [plugins](#) for an UNKNOWN_CMD command and execute it if found.

Warnings

When you define an alias in the command line (i.e., without using the [/R](#) option), variables and functions not protected by back quotes or doubled % signs are immediately evaluated, and the result becomes part of the alias value.

Syntax errors in an alias are not detected until the alias is executed.

Options:

- /G** Switch from a local to a global alias list.
- /L** Switch from a global to a local alias list.
- /O** Don't overwrite existing values (only valid in combination with [/R](#)).
- /P** This option is only effective when ALIAS is used to display existing definitions. It pauses the display after each page and waits for a keystroke before continuing (see [Page and File Prompts](#)).
- /R** This option loads an alias list from a file. The format of the file is the same as that of the ALIAS display:

name=value

where **name** is the *name* of the alias and **value** is its *value*. You can use an equal sign = or space to separate **name** and **value**. Back quotes are not required around the value. Variables and functions referenced in the definitions remain in the definitions, to be evaluated each time the alias is executed. You can add comments to the file by starting each comment line with a colon :. You can load multiple files with one **ALIAS /R** command by placing the names on the command line, separated by spaces:

```
alias /r alias1.lst alias2.lst
```

Each definition in an **ALIAS /R** file can be up to 32,767 characters long. The definitions can span multiple lines in the file if each line of the definition, except the last, is terminated with an [escape character](#).

ALIAS /R will read from stdin if no filename is specified and input is redirected:

```
alias /r <
```

- /Z** Overwrite the existing alias list with the contents of the specified file (must be used with [/R](#)). ALIAS /R /Z is 20x faster than an ALIAS /R, because it doesn't have to check for existing aliases and append new aliases to the end of the list.

4.3.7 ASSOC

Purpose: Modify or display relationships between file extensions and file types stored in the Windows registry

Format: ASSOC [/P /R [file...]] | [.ext=[filetype]] /U]

file	One or more input files to read association definitions from.
.ext	The file extension whose file type you want to display or set.
filetype	A file type stored in the Windows registry.

<u>/P(ause)</u>	<u>/U(ser)</u>
<u>/R(read)</u>	

See also: [FTYPE](#), and [Executable Extensions](#).

Usage:

ASSOC allows you to create, modify, or display associations between file extensions and file types stored in the Windows registry.

ASSOC manages Windows file associations stored under the registry handle HKEY_CLASSES_ROOT, and discussed in more detail under [Windows File Associations](#). If you are not familiar with file associations be sure to read about them before using ASSOC.

If you invoke ASSOC with no parameters, it will display the current associations. If you include a **.ext**, with no equal sign or **filetype**, ASSOC will display the current association for that extension.

If you include the equal sign and **filetype**, ASSOC will create or update the association for extension **.ext** to refer to the specified file type. The valid file types depend on the contents of your Windows registry. See the [FTYPE](#) command or your Windows documentation for additional details.

ASSOC cannot delete an extension from the registry. However, you can create a similar effect by associating the extension with an empty file type using **ASSOC .ext=**, without the **filetype** parameter.

ASSOC should be used with caution, and only after backing up the registry. Improper changes to file associations can prevent applications and / or the operating system from working properly.

Options:

/P Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).

/R This option loads an association list from a file. The format of the file is the same as that of the ASSOC display:

.ext=filetype

where **.ext** is an extension, which is to be associated with **filetype**.

You can load multiple files with one ASSOC /R command by placing the names on the command line, separated by spaces:

```
assoc /r assoc1.lst assoc2.lst
```

You can insert comments in the file by prefixing the line with a colon (:).

ASSOC /R will read from stdin if no filename is specified and input is redirected.

/U Display or set the association in HKCU\Software\Classes.

4.3.8 ASSOCIATE

Purpose: Display, create, or delete file / command associations.

Format: ASSOCIATE [/D /F /P /R [file...]] /U [.ext[=[command]]]

file One or more input files to read association definitions from.

.ext The file extension whose associated command you want to display or set.

command The executable command to run for the specified file extension

/D(delete)

/R(read)

/F(force)

/U(user)

/P(pause)

See also: [ASSOC](#), [FTYPE](#), and [Executable Extensions](#).

Usage:

If you are running Windows Vista or above, you must be running in an elevated session to use ASSOCIATE (unless you're using the /U option).

ASSOCIATE combines the ASSOC and FTYPE commands. It allows you to create, modify, or display associations between file extensions and commands types stored in the Windows registry.

ASSOCIATE manages Windows file associations stored under the registry handle HKEY_CLASSES_ROOT (or HKEY_CLASSES_USER), and discussed in more detail under [Windows File Associations](#). If you are not familiar with file associations be sure to read about them before using ASSOCIATE.

If you invoke ASSOCIATE with no parameters, it will display the current associations. If you include a **.ext**, with no equal sign or **command**, ASSOCIATE will display the command associated with that extension.

If you include the equal sign and **command**, ASSOCIATE will create or update the association for extension **.ext** to refer to the specified command.

ASSOCIATE should be used with caution, and only after backing up the registry. Improper changes to file associations can prevent applications and / or the operating system from working properly.

Options:

/D Delete the association for the specified **.ext**.

/F Force an overwrite of an existing association.

/P Wait for a key to be pressed after each screen page before continuing the display. Your

options at the prompt are explained in detail under [Page and File Prompts](#). (Only useful when running ASSOCIATE with no arguments.)

- /R** This option loads an association list from a file. The format of the file is the same as that of the ASSOCIATE display:

```
.ext=command
```

where **.ext** is an extension, which is to be associated with **command**.

You can load multiple files with one ASSOCIATE /R command by placing the names on the command line, separated by spaces:

```
associate /r assoc1.lst assoc2.lst
```

You can insert comments in the file by prefixing the line with a colon (:).

ASSOCIATE /R will read from stdin if no filename is specified and input is redirected.

- /U** Display or set the association in HKCU\Software\Classes.

4.3.9 ATTRIB

Purpose: Change or view file and subdirectory attributes

Format: ATTRIB [/A:[[-+]rsa] /D /E /I"text" /L /N[EJ] /O:[-]adegrnstu /P /Q /S[[+]n]] [+|- [AHIORSTVX]] [@file] *files* ...

files A file, directory, or list of files or directories to process.

@file A text file containing the names of the files to process, one per line (see [@file lists](#) for details).

/A: (Attribute select)	/N(o)
/D(irectories)	/O:....(order)
/E (No error messages)	/P(ause)
/I"text" (match description)	/Q(uiet)
/L (symbolic Link)	/S(ubdirectories)

Attribute flags:

Clear	Set	Attribute affected
-A	+A	archive
-C	+C	compressed
-H	+H	hidden
-I	+I	not content indexed
-O	+O	offline
-R	+R	read-only
-S	+S	system
-T	+T	temporary
-V	+V	integrity (Windows 8 ReFS only)
-X	+X	no_scrub_data (Windows 8 ReFS only)

File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and

[include lists](#). Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Usage:

Every file and subdirectory has attributes that can be turned on (set) or turned off (cleared): **Archive**, **Hidden**, **Not content indexed**, **Offline**, **Read-only**, **System**, and **Temporary**. For details on the meaning of each attribute, see [File Attributes](#).

The ATTRIB command lets you view, set, or clear attributes for any file, group of files, or subdirectory.

You can view file attributes by entering ATTRIB without specifying new attributes (*i.e.*, without the **[+|-]
[AHIORST]** part of the format), or with the [DIR /T](#) command.

The primary use of ATTRIB is to set attributes. For example, you can set the read-only and hidden attributes for the file *MEMO*:

```
attrib +rh memo
```

Attribute options apply to the file(s) that follow the options on the ATTRIB command line. The example below shows how to set different attributes on different files with a single command. It sets the archive attribute for all *.TXT* files, then sets the system attribute and clears the archive attribute for *TEST.EXE*:

```
attrib +a *.txt +s -a test.exe
```

When you use ATTRIB on an LFN drive, you must double quote any file names which contain white space or special characters.

To change directory attributes, use the **/D** switch. If you give ATTRIB a directory name instead of a file name, and omit **/D**, it will append "*****" to the end of the name and act on all files in that directory, rather than acting on the directory itself.

NTFS also supports **D** (subcategory), **V** (virtualized), **E** (encrypted), **J** or **L** (junction / symbolic link) and **P** (sparse file) attributes. These attributes will be displayed by ATTRIB, but cannot be altered; they are designed to be controlled only by Windows.

ATTRIB will ignore underlines in the new attribute (the **[+|-]
[ADHIORST]** part of the command). For example, ATTRIB sees these 2 commands as identical:

```
attrib +a filename  
attrib +__A__ filename
```

This allows you to use a string of attributes from either the [@ATTRIB](#) variable function or from ATTRIB itself (both of which use underscores to represent attributes that are not set) and send that string back to ATTRIB to set attributes for other files. For example, to clear the attributes of *FILE2* and then set its attributes to match those of *FILE1*:

```
attrib -arhs file2 & attrib +%@attrib[file1] file2
```

When ATTRIB encounters a **+D** or **-D** in the attribute string it treats it as equivalent to the **/D** switch, and allows modification of the attributes of a directory. When combined with [@ATTRIB](#), or with ATTRIB's output, both of which return a **D** to signify a directory, this feature allows you to transfer attributes from one directory to another. For example, to clear the attributes of all files and directories beginning with *ABC* and then set their attributes to match those of *FILE1* (enter this on one line):

```
attrib -arhs abc* & attrib +%@attrib[file1] abc*
```

ATTRIB sets three internal variables:

%_attrib_dirs	The number of directories modified
%_attrib_files	The number of files modified
%_attrib_errors	The number of errors

Options:

- /= Display the ATTRIB command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A: Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Warning: the colon after /A is not optional.

This switch specifies which files to select, not which attributes to set. For example, to remove the archive attribute from all hidden files, you could use this command:

```
attrib /a:h -a *
```

Do not use /A: with [@file lists](#). See [@file lists](#) for details.

You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

- /D If you use the /D option, ATTRIB will modify the attributes of directories in addition to files (yes, you can have a hidden directory):

```
attrib /d +h c:\mydir
```

If you use a directory name instead of a file name, and omit /D, ATTRIB will append "*" to the end of the name and act on all files in that directory, rather than acting on the directory itself.

- /E Suppress all non-fatal error messages, such as "File Not Found." Fatal error messages, such as "Drive not ready," will still be displayed. This option is most useful in batch files and aliases, and when recursing through the directory hierarchy, where many directories have no files matching your selection criteria.

- /I"text" Select files by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must immediately follow the /I, with no intervening spaces. You can select all filenames that have a description with /I"[?]*", or all filenames that do not have a description with /I"\"". Do not use /I with [@file lists](#). See [@file lists](#) for details.

- /L Set or display the attributes of the symbolic link versus the target of the symbolic link.

- /N Do everything except actually change the attributes. This option is useful for testing what the result of a complex ATTRIB command will be. (Not available in **TCC/LE**.)

A **/N** with one of the following arguments has an alternate meaning:

- e** Don't display errors.
- j** Skip junctions (when used with **/S**)

/O:.... Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

/P Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#)

/Q This option turns off ATTRIB's normal screen output. It is most useful in batch files.

/S If you use the **/S** option, the ATTRIB command will be applied to all matching files in the current or named directory and all of its subdirectories. Do not use **/S** with **@file lists**; see [@file lists](#) for details.

If you specify a number after the **/S**, ATTRIB will limit the subdirectory recursion to that number. For example, if you have a directory tree "**\a\b\c\d\le**", **/S2** will only affect the "a", "b", and "c" directories.

If you specify a **+** followed by a number after the **/S**, ATTRIB will not modify any file attributes until it gets to that depth in the subdirectory tree. For example, if you have a directory tree **\a\b\c\d\le**, **/S+2** will not modify anything in **\a** or **\a\b**. (Not available in **TCC/LE**.)

4.3.10 BATCOMP

Purpose: Compress and encrypt batch files. See [Batch File Compression](#) for additional details.

Format: BATCOMP [**/O** | **/Q**] InputFile OutputFile

- | | |
|-------------------|--|
| InputFile | A file to compress and/or encrypt. |
| OutputFile | A file to hold the output from the command |

[**/O\(overwrite\)**](#)

[**/Q\(uiet\)**](#)

File Selection

The input file must be specified explicitly (no wildcards).

Usage:

BATCOMP is a batch file compressor and optionally allows for simple key-based encryption.

If you do not specify an extension for **OutputFile**, it defaults to **BTM**. If **OutputFile** already exists it will not be overwritten unless **/O** is used.

The output BTM file will not be legible, but it will run under **TCC**. The behavior and performance of the file should be the same as if it were run in its original source form as a **.BTM** file.

Compression is not effective for very small files and may even result in a larger file.

Options:

/O Forces overwriting of an existing **OutputFile**.

/Q Suppresses all progress reports (stdout). Errors (stderr) are still shown.

4.3.11 BDEBUGGER / IDE

Purpose: Calls the **Take Command** IDE / batch debugger (IDE.EXE)

Format: BDEBUGGER [/C] batchfilename [parameters]
or
IDE [/C] file...

batchfilename Full name of the batch file to debug.

parameters Parameters for the batch file

file File(s) to edit

[/C\(reate new batch file\)](#)

Usage:

BDEBUGGER and IDE open a development environment in which file can be edited and (in the case of batch files) debugged. The difference between the two commands is that BDEBUGGER assumes that you're trying to edit & debug a batch file. BDEBUGGER will add a missing .BTM, .CMD, or .BAT extension on the filename, and it assumes that any arguments following the batch file name are batch file parameters. The IDE command is intended as a generic file editor (it supports syntax colorization for several scripting languages, including .BAT, .BTM, and .CMD batch files, Perl, Python, Ruby, and Tcl). Each argument on the command line is assumed to be a filename to be opened in a separate tab window.

The BDEBUGGER / IDE window includes a slider control on the lower right corner of the status bar to control the transparency level.

The edit window toolbar (which is configurable by clicking on the rightmost down arrow), has a number of icons to control debugging. Each has a tooltip for quick reference:

New

Create a new batch file in a new tab window.

Open	Open an existing batch file in a new tab window.
Save	Save the current batch file.
Print	Print the current batch file.
Cut	Copy the highlighted selection to the clipboard and delete it from the file.
Copy	Copy the highlighted selection to the clipboard.
Paste	Copy the contents of the clipboard to the current cursor location.
Delete	Delete the highlighted selection.
Undo	Undo the last edit.
Redo	Restore the last Undo.
Find	Search for text.
Batch Arguments	New batch file arguments. The text will be parsed into %1 - %n batch arguments and used when the batch file is debugged.
Start Debugging	Starts the debugger. The cursor will be placed on the first line.
Pause Debugging	Pause execution at the next line.
Stop Debugging	Stops the debugger.
Step Into	Execute the current line.
Step Over	Execute the current line but disable the debugger during a CALL or GOSUB.
Run to Breakpoint	Execute the batch file, stopping at the next breakpoint.
Toggle Breakpoint	Sets or turns off a breakpoint on the current line.
Clear Breakpoints	Clear all breakpoints in the current batch file.
File Properties	Displays information on the current batch file.
Start New Shell	Start another copy of TCC (this is useful if you need to perform some tasks while debugging a file.)
Help	Display the online help.

You can get help for the currently selected (highlighted) command / variable / function by pressing Ctrl-F1, or right-clicking the mouse and selecting **Help** from the context menu.

If you press **Ctrl-C** or **Ctrl-Break** while debugging, you will see the prompt:

Cancel batch job *filename* (Y/N/A/D) :

Pressing **D** will return you to single-step mode in the debugger. (This allows you to interrupt a **run-to-breakpoint** without terminating the debugger and batch file.)

You can change the line to be executed next when in debugging mode by moving the caret to the line and either right clicking & selecting "Jump to This Line" or by pressing Ctrl-Shift-F11. Note that if you attempt to jump into or out of a DO loop or IFF block, bad things will happen!

Alt-F11 will invoke the Evaluate Expression dialog. Alt-Shift-F11 will invoke the Evaluate Expression dialog for the current selection, or if no text is selected, for the current line.

The IDE includes tabbed windows to display and/or modify the breakpoint list, watch list, aliases, batch variables, environment variables, and user functions. The variable windows also have a toolbar, with the following buttons:

New	Restore the original values for the list (alias, variable, environment, or function)
Open	Add the contents of a file to the list
Save	Save the current list to a file
Reset	Replace the original values with the modified list
Print	Print the current list
Cut	Copy the highlighted selection to the clipboard and delete it from the file

Copy	Copy the highlighted selection to the clipboard
Paste	Copy the contents of the clipboard to the current cursor location
Delete	Delete the highlighted selection (or the character at the cursor location if no selection)
Undo	Undo the last edit
Redo	Restore the last Undo
Find	Search for text
Help	Display the online help

The environment variables window displays any variables changed since the last command in red. You can specify variables to exclude from the environment variable window with the DebugVariableExclude variable. For example, to suppress the display of the processor and user variables:

```
set DebugVariableExclude=proc*;user*
```

Note that this option doesn't affect the existence of the variables, just whether they're displayed in the environment variable window.

Breakpoints

You can set a breakpoint either through the menu, the toolbar, or by moving the mouse cursor to the left margin of a line and left-clicking. You can only set a breakpoint on an executable line (i.e., not on a blank line, comment, label, etc.).

You can define conditional breakpoints by specifying the number of iterations before the breakpoint is triggered, and/or define a [conditional expression](#) that must be true before the breakpoint is triggered. After setting the breakpoint, enter the conditions either by right-clicking on the breakpoint and selecting "Break >=" (to set the minimum number of iterations), or "Condition" (to set the conditional expression). You can also select the Breakpoint window and double-click on the "Break >=" or "Condition" columns to edit or modify the conditions.

You can temporarily disable a breakpoint (without deleting it) by right-clicking on the breakpoint and selecting "Enable/Disable Breakpoint". You can disable all breakpoints by clicking on the Debug menu and selecting "Disable All Breakpoints".

Watch Window

The Watch window allows you to monitor environment variables, internal variables, variable functions, and user-defined functions, or to pause execution when a specified condition is met. The Watch window appears at the bottom of the debugger window. Enter the variable name or expression in the left column; the debugger will automatically display the current value in the right column. You can also add a variable to the Watch window by selecting it in the main debugger window, then clicking the right mouse button and selecting "Add to Watch". If the string in the left column is a single argument, it is assumed to be a variable name. Otherwise, it is assumed to be an expression. Expressions can be anything that IF can evaluate; for example:

```
%i = 3
ERRORLEVEL GT 12
```

Note that expressions require variable names to be prefixed by a %. If you're entering a single variable argument to monitor, do not use a %.

If you right click on the first column in the Watch window, the debugger will display an environment

variable listbox. Select an entry to have it added to the watch list.

When the value of a monitored variable changes, the Watch window will change the text color to red.

Margins

There are three possible margins on the left of the edit window:

- The line number (selectable by the "Options / Display Line Numbers" menu option).
- The Breakpoint margin (left click in this margin to set a breakpoint on this line).
- The Fold margin (selectable by the "Options / Display Fold Margin" menu option), which will display a - for blocks that can be collapsed to a single line (DO, IFF, and SWITCH commands, and command groups). When a block is collapsed, the Fold margin will display a +. Left clicking in the Fold margin will toggle the fold state.

Edit Windows

The text processing commands available in the IDE edit windows are listed below. The text commands can be classified into general categories:

- [Caret commands](#)
- [Edit commands](#)
- [Mark / Clipboard commands](#)
- [Search commands](#)
- [File commands](#)
- [Bookmark commands](#)
- [Breakpoint commands](#)
- [Expression evaluation commands](#)

• Caret commands

Right

This command will move the caret one character to the right. When the caret is on the last position of the current line it is moved to the first position of the next line.

Shift-Right

In addition to the caret movement this command will also extend the current selection to the new caret position.

Left

This command will move the caret one character to the left. When the caret is on the first position of the current line it is moved to the last position of the previous line.

Shift-Left

In addition to the caret movement, this will also extend the current selection to the new position.

Up

This command will move the caret one line up. The caret column position will be set as close to its previous column position as possible.

Shift-Up

In addition to the caret movement this command will also extend the current selection to the new position.

Down

This command will move the caret one line down. The caret column position will be set as close to its previous column position as possible. When the caret is on the last line but not on the last column it will be moved to the last column.

Shift-Down

In addition to the caret movement this command will also extend the current selection to the new position.

End

This command will move the caret to the end of the line it is currently on. If the caret is already at the end nothing happens.

Shift-End

In addition to the caret movement this command will also extend the current selection to the new position.

Home	This command will move the caret to the start of the line it is currently on. If the caret is already at the start nothing happens.
Shift-Home	In addition to the caret movement this command will also extend the current selection to the new position.
Ctrl-Right	This command will move in one of the following ways: <ul style="list-style-type: none"> • When the caret is located on a delimiter character the caret is moved right until the first non-delimiter is found. • When the caret is located on a non-delimiter character the caret is moved to the next delimiter character. • When the caret is located on the last word or delimiter of the current line the caret is moved to the first word or delimiter of the next line.
Ctrl-Shift-Right	In addition to the caret movement this command will also extend the current selection to the new caret position.
Ctrl-Left	This command will move in one of the following ways: <ul style="list-style-type: none"> • When the caret is located on a delimiter character the caret is moved to the start of the previous word. • When the caret is located on a non-delimiter character and not on a white-space character the caret is moved to the start of the current word. • When the caret is located on the start of the first word, delimiters or white-space of the current line the caret is moved to the start of the last word or delimiters of the previous line.
Ctrl-Shift-Left	In addition to the caret movement this command will also extend the current selection to the new position.
Ctrl-Home	This command will move the caret to the beginning of the text. When the caret is already at this location nothing happens.
Ctrl-Shift-Home	In addition to the caret movement this command will also extend the current selection to the new position.
Ctrl-End	This command will move the caret to the end of the text. When the caret is already at this location nothing happens.
Ctrl-Shift-End	In addition to the caret movement this command will also extend the current selection to the new position.
PgUp	This command will move the caret one view up when it is located on the top line currently in the view. When the caret is not located on the top line of the view, it will be moved there.
Shift-PgUp	In addition to the caret movement this command will also extend the current selection to the new position.
PgDn	This command will move the caret one view down when it is located on the bottom line currently in the view. When the caret is not located on the bottom line of the view, it will be moved there.
Shift-PgDn	In addition to the caret movement, this command will also extend the current selection to the new position.

- **Edit commands**

Ctrl-Z	This command will undo the last change made to the edit control contents. You can undo any number of changes made to the control contents up to the maximum number of undo/redo hops.
Ctrl-Y	This command will redo the last change you have undone. You can re-do any number of changes up to the number of changes undone.
Backspace	This command will remove the character to the left of the caret. When the caret is located at the start of the line, the characters right of the caret are appended to the previous line and the caret is moved to be positioned between the old line contents and the appended characters.
Delete	This command removes the character to the right of the caret. When there are

	no characters to the right of the caret, the contents of the next line is appended to the current line.
Return	This command will split the current line and create a new line of the characters, if any, right of the caret. The caret is moved to the start of the newly created line.
Ctrl-Delete	When the caret is located on a word, this command will delete all characters in the word right of the caret position.
Ctrl-Backspace	When the caret is located on a word, this command will delete all characters in the word left of the caret position.
Tab	This command does one of the two following things: <ul style="list-style-type: none"> • When there is a valid text selection, this command will indent the lines covered by the selection right by one tab-stop. • When there is no text selection, a tab is inserted at the current caret position.
Shift-Tab	When there is a valid text selection, this command will indent the lines covered by the selection left by one tab-stop.
Shift-Ctrl-U	When there is a valid selection, this command will convert all lower-case characters in the selection to upper-case characters. If there is no valid selection, nothing happens.
Ctrl-U	When there is a valid selection, this command will convert all upper-case characters in the selection to lower-case characters. If there is no valid selection, nothing happens.
Ins	This command will toggle the current editing mode between overwrite and insert.

• Mark / Clipboard commands

Ctrl-A	This command will select all the text.
Ctrl-V	This command will, when there is text present in the clipboard, paste the clipboard contents at the current position.
Ctrl-C	This command will, when there is a selection, copy the selected text to the clipboard.
Ctrl-X	This command will, when there is a selection, copy the selected text to the clipboard and remove the selection from the text.

• Search commands

Ctrl-F3	This command will find the next occurrence of the word under the caret. When the next occurrence is found, it is selected.
F3	This command will find the next occurrence of the current search pattern. When the search pattern is found, it is selected.
Shift-F3	This command will find the previous occurrence of the current search pattern. When the search pattern is found, it is selected.
Ctrl-G	This command will show the <i>goto</i> dialog.
Ctrl-F	This command will show the <i>find</i> dialog.
Ctrl-H	This command will show the <i>replace</i> dialog.

• File commands

Ctrl-N	Open a new file in a new tab window.
Ctrl-O	Open an existing file in a new tab window.
Ctrl-W	Close all files.
Ctrl-S	This command will save the current file.
Ctrl-Shift-S	Save all files.

Ctrl-P	This command will open the print dialog.
Ctrl-I	Display the properties for the current file.
Alt-F4	Exit the debugger.

- **Bookmark commands**

Ctrl-F2	This command will clear the bookmark on the current line if it is set, or set the bookmark if it is cleared.
Shift-Ctrl-F2	This command will clear all bookmarks.
F2	This command will place the caret on the next line which has a bookmark set. When there is no next line with a bookmark, the text is searched starting at the first line.
Shift-F2	This command will place the caret on the previous line which has a bookmark set. When there is no previous line with a bookmark, the text is searched from the last line up again.

- **Breakpoint commands**

Ctrl-F9	This command will toggle a breakpoint on the current line.
Ctrl-Shift-F9	This command will clear all breakpoints.

- **Expression evaluation commands**

Alt-F11	Invoke the Evaluate Expression dialog.
Alt-Shift-F11	Invoke the Evaluate Expression dialog for the current selection. If no text is selected, evaluate the current line.

You can select the result and copy it to the clipboard.

Options:

/C If the specified batch file doesn't exist, create it without prompting.

4.3.12 BEEP

Purpose: Beep the speaker or play simple music

Format: BEEP [frequency duration ...] [asterisk | exclamation | hand | question | ok]

frequency	The beep frequency in Hertz (cycles per second).
duration	The beep length in 1/18th second intervals.
asterisk	Plays the system default "asterisk" sound.
exclamation	Plays the system default "exclamation" sound.
hand	Plays the system default "hand" sound.
question	Plays the system default "question" sound.
ok	Plays the system default "ok" sound.

See also: the [Length](#) and [Frequency](#) configuration options.

Usage:

BEEP generates a sound through your computer's speaker. You can use it in batch files to signal that

an operation has been completed, or that the computer needs attention.

Because 64-bit versions of Windows do not support playing sounds through the Windows Beep API, **TCC x64** uses DirectSound for BEEP.

Because BEEP allows you to specify the frequency and duration of the sound, you can also use it to play simple music or to create different kinds of signals for the user.

You can include as many frequency and duration pairs as you wish. No sound will be generated for frequencies less than 20 Hz, allowing you to use BEEP as a way to create short delays. The default value for *frequency* is 440 Hz; the default value for *duration* is 2.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

This batch file fragment runs a program called *DEMO*, then plays a few notes and waits for you to press a key:

```
demo
beep 440 4 600 2 1040 6
pause Finished with the demo - hit a key...
```

The following table gives the *frequency* values for a five octave range (middle C is 262 Hz):

C	131	262	523	1046	2096
C# / Db	139	277	554	1108	2217
D	147	294	587	1175	2349
D# / Eb	156	311	622	1244	2489
E	165	330	659	1318	2637
F	175	349	698	1397	2794
F# / Gb	185	370	740	1480	2960
G	196	392	784	1568	3136
G# / Ab	208	415	831	1664	3322
A	220	440	880	1760	3520
A# / Bb	233	466	932	1866	3729
B	248	494	988	1973	3951

4.3.13 BREAK

Purpose: Enable or disable Ctrl-C and Ctrl-Break

Format: BREAK [ON | OFF]

Usage:

BREAK OFF will disable all **Ctrl-C** and **Ctrl-Break** handling in **TCC** (though not necessarily in child processes). In CMD, BREAK OFF doesn't actually do anything, so setting it in **TCC** will introduce a possible incompatibility with existing batch files.

4.3.14 BREAKPOINT

Purpose: Set a batch debugger breakpoint on the current line

Format: BREAKPOINT

Usage:

If the batch debugger is active, BREAKPOINT sets a breakpoint on the current line, stopping a "Step Out" sequence. If the batch debugger is not active, BREAKPOINT is ignored.

4.3.15 CALL

Purpose: Execute one batch file from within another.

Format: CALL file | :label [p1 [p2 ...]]

file	The batch file to execute.
:label	A label in the current batch file.
p1, p2,...	Parameters for the batch file or subroutine

See also: [CANCEL](#) and [QUIT](#).

Usage:

Calling other batch files

CALL allows batch files to call other batch files (batch file nesting). The calling batch file is suspended while the called (second) batch file runs. When the second batch file finishes (without executing the CANCEL command), execution of the original batch file resumes at the next command.

WARNING! If you execute a batch file from inside another batch file without using CALL, the original batch file is terminated before the other one starts. This method of invoking a batch file from another is usually referred to as chaining. Note that if the batch file *A.BTM* uses **CALL B**, and *B.BTM* chains to the batch file *C.BTM*, on exit from *C.BTM* (without executing a [CANCEL](#) command) processing of batch file *A.BTM* is resumed as if it had used **CALL C**.

File *A.BTM*:

```
...
call b
echo xxx
```

File *B.BTM*:

```
...
C
```

File *C.BTM*:

```
...
quit
```

In the example above, after execution of the [QUIT](#) command in *C.BTM* the ECHO xxx command in

A. *BTM* is executed next.

The following batch file fragment compares an input line to *wp* and calls another batch file if it matches:

```
input Enter your choice: %%option
if "%option" == "wp" call wp.bat
```

Batch files may be nested up to 32 levels deep.

The current ECHO state is inherited by a called batch file.

The called batch file should always either return (by executing its last line, or by using the [QUIT](#) command), or it should terminate batch file processing with [CANCEL](#). Do not restart or CALL the original batch file from within the called file as this may cause an infinite loop or a stack overflow.

Calling a label

To provide compatibility with CMD, which does not support the [GOSUB](#) command for subroutines in the same batch file, you may create a subroutine starting with a label and terminated by any of the following:

- the end of the batch file
- [QUIT](#)
- [EXIT](#)
- [CANCEL](#)

Note that the last two do NOT return control to the CALL command. Do not use the [RETURN](#) command!

Parameters passed to the subroutine are accessible as *%1*, *%2*, etc., in the same manner as in a batch file.

Exit code

CALL returns an exit code which matches the batch file return code. You can test this exit code with [conditional commands](#) (**&&** and **||**).

See also [GOSUB](#) and [user-defined functions](#).

4.3.16 CANCEL

Purpose: Terminate batch file processing

Format: CANCEL [value]

value The numeric exit code to return to **TCC**.

See also: [CALL](#) and [QUIT](#).

Usage:

The CANCEL command ends all batch file processing, regardless of the batch file nesting level. Use [QUIT](#) to end a nested batch file and return to the previous batch file.

You can CANCEL at any point in a batch file. If CANCEL is used from within an alias it will end execution of both the alias and any batch files which are running at the time.

The following batch file fragment compares an input line to "end" and terminates all batch file processing if it matches:

```
input Enter your choice: %%option
if "%option" == "end" cancel
```

If you specify a **value**, CANCEL will set the ERRORLEVEL or exit code to that value (see the [IF](#) command, and the [%?](#) variable).

4.3.17 CD / CHDIR

Purpose: Display or change the current directory

Format: CD [/D /N /X] [path | -]

path The directory to change to, optionally including a drive letter

/D(rive)	/X(exclude)
/N(o extended search)	

See also: [CDD](#), [MD](#), [PUSHD](#), [RD](#), and [Directory Navigation](#).

Internet: Can be used with [FTP Servers](#).

Usage:

CD and CHDIR are synonyms. You can use either one.

CD lets you navigate through a drive's subdirectory structure by changing the current working directory. If you enter CD and a directory name, the named directory becomes the new current directory. For example, to change to the subdirectory C:\FINANCE\MYFILES:

```
[c:\] cd \finance\myfiles
[c:\finance\myfiles]
```

Every disk drive on the system has its own current directory. Specifying both a drive and a directory in the CD command will change the current directory on the specified drive, but will not change the default drive (unless you use the [/D](#) option). For example, to change the default directory on drive A:

```
[c:\] cd a:\utility
[c:\]
```

Notice that this command does not change to drive A:. Use the [/D](#) option, or preferably the [CDD](#) command to change the current drive and directory at the same time.

If **path** contains white space or special characters (which is valid only for an LFN drive), you must enclose it in double quotes.

If **path** begins with a ~ (tilde), CD will substitute to the user's home directory, as defined by HOME in the environment. (If HOME doesn't exist, TCC will look for %HOMEDRIVE + HOME PATH.)

You can change to the parent directory with **CD ..**; you can also go up one additional directory level with

each additional ... For example, **CD** will go up three levels in the directory tree (see [Extended Parent Directory Names](#)). You can move to a sibling directory (one that branches from the same parent directory as the current subdirectory) with a command like **CD ..\newdir**.

If you enter CD with no parameter or with only a disk drive name, it will display the current directory on the default or named drive.

If CD cannot change to the directory you have specified it will attempt to search the [CDPATH](#) and the [extended directory search](#) database in order to find a matching directory and switch to it. You can disable this default extended search with [/N](#). You can also use wildcards in **path** to force an extended directory search. Read the section on [Directory Navigation](#) for complete details on these and other directory navigation features.

If the TCMD.INI directive [EverythingSearch](#) is set, CD will use **Everything Search** (free at <http://www voidtools com>) instead of JPSTREE.IDX for fuzzy directory searches. Everything Search is slightly faster, but will only work on local NTFS drives. Setting EverythingSearch is the equivalent of setting FuzzyCD=3 (*name*). You must download and install **Everything Search** yourself; it is not included in the **Take Command** distribution.

CD saves the current directory before changing to a new directory. You can switch back to the previous directory by entering **CD -** (There must be a space between the CD command and the hyphen.) You can switch back and forth between two directories by repeatedly entering **CD -**. The saved directory is the same for both the CD and [CDD](#) commands. Drive changes and [automatic directory changes](#) also modify the saved directory, so you can use **CD -** to return to a directory that you exited with an automatic directory change. **TCC** recognizes a single hyphen on the command line as an internal alias for **CDD -**.

Directory changes made with CD are recorded in the directory history list and can be displayed in the [directory history window](#), which allows you to return quickly to a recently-used directory.

You can also use CD to display or change the current directory on an [FTP server](#) opened with [IFTP](#). For example:

```
cd ftp:  
ftp://ftp.microsoft.com/  
  
cd ftp:/pub
```

Note: FTP directory changes use neither the [CDPATH feature](#) nor the [Extended Directory Searches](#) database.

CD never changes the default drive, unless the [/D](#) option is specified. If you change directories on one drive, switch to another drive, and then enter **CD -**, the directory will be restored on the first drive but the current drive will not be changed.

Options:

- /D** Changes the current drive as well as directory. This option is included only for compatibility with the same option available in some versions of CMD. In most cases you should use [CDD](#), which performs the same function.
- /N** Skips the standard [extended directory search](#) when the directory is not found. This option is useful in batch files to force an error (rather than an extended search) if a directory is not

found.

/X Don't save the current directory to the Directory History list.

4.3.18 CDD

Purpose: Change the current disk drive and directory

Format: CDD [/A /D[drive ...] /N[J] /S[n][drive ...] /U[n][drive ...] /X] [path | -]

path The name of the directory (or drive and directory) to change to.

drive A drive or list of drives to include in the extended directory search database.

/A(ll drives)

/T (Also change Folders directory)

/D(elete from JPSTREE.IDX)

/TO Only change Folders directory

/N(o extended search)

/U(pdate tree)

/NJ (Skip junctions)

/X(exclude from directory history)

/S(earch tree)

See also: [CD](#), [MD](#), [PUSHD](#), [RD](#), and [Directory Navigation](#).

Usage:

CDD is similar to the [CD](#) command, except that it also changes the default disk drive if one is specified. For example, to change from the root directory on drive A to the subdirectory C:\WP:

```
[a:\] cdd c:\wp
[c:\wp]
```

If no drive / path argument is supplied, CDD displays the current drive and directory.

CDD can also be used to create and update the [Extended Directory Search](#) database (*JPSTREE.IDX*).

If **path** begins with a ~ (tilde), CD will substitute to the user's home directory, as defined by HOME in the environment. (If HOME doesn't exist, TCC will look for %HOMEDRIVE + HOME PATH.) (Not supported in TCC/LE.)

You can change to the parent directory with **CDD ..**; you can also go up one additional directory level with each additional **[.]**. For example, **CDD** will go up three levels in the directory tree.

CDD can also change to a network drive and directory specified with a UNC name (see [File Systems](#) for details).

When you use CDD to change to a directory on an LFN drive, you must quote the **path** name if it contains white space or special characters.

If CDD cannot change to the directory you have specified it will first search the [CDPATH](#), then the [extended directory search](#) database in order to find a matching directory and switch to it. You can disable this default extended search with **/N**. You can also use wildcards in the **path** to force an extended directory search. Read the section on [Directory Navigation](#) for complete details on these and other directory navigation features.

If the TCMD.INI directive [EverythingSearch](#) is set, CDD will use **Everything Search** (free at <http://>

www.voidtools.com) instead of JPSTREE.IDX for fuzzy directory searches. Everything Search is slightly faster, but will only work on local NTFS drives. Setting EverythingSearch is the equivalent of setting FuzzyCD=3 (*name*). You must download and install **Everything Search** yourself; it is not included in the **Take Command** distribution.

CDD saves the current drive and directory before changing to a new directory. You can switch back to the previous drive and directory by entering **CDD -**. (There must be a space between the CDD command and the hyphen.) You can switch back and forth between two drives and directories by repeatedly entering **CDD -**. The saved directory is the same for both the CD and CDD commands. Drive changes and [automatic directory changes](#) also modify the saved directory, so you can use **CDD -** to return to a directory that you exited with a drive change or an automatic directory change. **TCC** recognizes a single hyphen on the command line as an internal alias for **CDD -**.

Directory changes made with CDD are recorded in the directory history list and can be displayed in the [directory history window](#), which allows you to return quickly to a recently-used directory.

Windows limits the permissible length of the full subdirectory name (see the [Directories and Subdirectories](#) topic for information on directory names).

When changing directories, **TCC** maintains the original case of each path element. This is necessary for a few programs which are case-sensitive in their use of directory names.

Options:

- /A** When CDD is used with this option, it displays the current directory on all drives from C: to the last drive in the system. You cannot move to a new drive and directory and use **/A** in the same command.
- /D** Removes the specified drives or directory trees from the [Extended Directory Search](#) database (*JPSTREE.IDX*). Uses the same syntax for drive and directory names as **/S**. For example, to delete the directories under *F:\MYDIR* from *JPSTREE.IDX*:

```
cdd /d f:\mydir
```

- /N** Skips the standard [extended directory search](#) when the directory is not found. This option is useful in batch files to force an error -- rather than an extended search -- if a directory is not found.
- /NJ** Skips junctions when indexing directories (see **/S**).
- /S** Builds or rebuilds the [Extended Directory Search](#) database (*JPSTREE.IDX*). You cannot move to a new drive and directory and use **/S** in the same command.

To include all local hard drives in the database, use the command:

```
cdd /s
```

To limit or add to the list of drives included in the database, list the drives and network volume names after the **/S** switch. For example, to include drives C, D, and E, and the sharename *\server\dir1*, use this command:

```
cdd /s c:\ d:\ e:\ \\server\dir1
```

All non-hidden directories on the listed drives will be indexed. CDD /S will also index the hidden directories if the [Complete Hidden Files](#) option is set. Each time you use **/S**, everything in the previous directory database is replaced by the new database that is created. To update the database see **/U** below.

You can index specific subdirectories rather than an entire drive. For example, to index all directories on drive C but only the MSSDK directory tree on drive D:

```
cdd /s c:\ d:\mssdk
```

If you specify a number after the **/S**, CDD will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\e", /S2 will only index the "a", "b", and "c" directories.

- /T** Also change the current directory in the **Take Command** Folders and List View windows.
- /TO** Change the current directory in the **Take Command** Folders and List View windows without changing the **TCC** current directory.
- /U** Updates the [Extended Directory Search](#) database (*JPSTREE.IDX*) with the specified drives and directories instead of rebuilding the whole directory database. Uses the same syntax for drive and directory names as **/S**. For example, to update the *D:\MSSDK* tree and all of drive E:

```
cdd /u d:\mssdk e:\
```

If you specify a number after the **/U**, CDD will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\e", /S2 will only update the "a", "b", and "c" directories.

Note: The [TREEEXCLUDE](#) variable can be used to specify which drives and directories should be ignored when updating the directory database.

- /X** Don't save the current directory to the Directory History list.

4.3.19 CHCP

Purpose: Display or change the current system code page

Format: CHCP [*n*]

n A system code page number.

Usage:

Code page switching allows you to select different character sets for language support.

If you enter CHCP without a number, the current code page is displayed.

```
chcp
Active code page: 437
```

If you enter CHCP plus a code page number, the code page is changed. For example, to set the code page to multilingual:

```
chcp 850
```

When you use CHCP under Windows it only affects the current process, and any new programs started from within that process; the active code page in other processes remains unchanged.

4.3.20 CLIPMONITOR

Purpose: Monitor changes in the Windows clipboard

Format: CLIPMONITOR [/C]
CLIPMONITOR n command

n Number of repetitions (or **FOREVER**)
command Command to execute when the clipboard is modified

[/C\(lear\)](#)

Usage:

If you don't enter any arguments, if CLIPMONITOR is active it will display the repeat count and the command.

The command line will be parsed and expanded before CLIPMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. CLIPMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

Options:

/C Remove the clipboard monitor.

4.3.21 CLS

Purpose: Clear the window and move the cursor to the upper left corner; optionally change the default display colors

Format: CLS [/C /S] [[BRIght] fg ON [BRIght] bg

fg The new foreground color
bg The new background color

[/C\(lear buffer\)](#)

[/S\(scroll buffer\)](#)

Usage:

CLS can be used to clear the window without changing colors, or to clear the window and change the colors simultaneously, or to clear the entire scrollback buffer. These two examples show how to clear the window to the default colors, and to bright white letters on a blue background:

```
cls  
cls bright white on blue
```

CLS is often used in batch files before displaying text.

See [Colors and Color Names](#) for details about colors.

Options:

- /C** Clears the entire scrollback buffer. If **/C** is not used, only the visible portion of the window is cleared.
- /S** Clear the screen by scrolling the buffer, rather than filling the screen with blanks (the default method). This saves the text on the screen into the scrollback buffer if it is larger than the visible window. This switch may not give the expected results when the buffer size is less than twice the window size.

4.3.22 COLOR

Purpose: Change the default display colors

Format: COLOR [BRIght] *fg* ON [BRIght] *bg*

- fg*** The new foreground color
- bg*** The new background color

See also: [CLS](#) and [Colors and Color Names](#) for details about using colors and the name and numeric codes for colors.

Usage:

COLOR is normally used in batch files before displaying text. For example, to set screen colors to bright white on blue, you can use this command:

```
color bright white on blue
```

TCC also supports the CMD syntax:

```
COLOR bf
```

In this syntax, **b** is a hexadecimal digit that specifies the background color and **f** is a hexadecimal digit that specifies the foreground color.

If you do not specify a new foreground and background color, COLOR will revert the display colors to those used when **TCC** was started (for compatibility with CMD).

If you have ANSI enabled and StdColors and/or InputColors set, they will override a COLOR command.

4.3.23 COPY

Purpose: Copy data between disks, directories, files, or physical hardware devices (such as your printer or serial port)

Format: COPY [/I"text"] [/A:... /C /CF /D /E /F /FTP:A /G /H /J /K /L /M /MD /N[dejnst] /O /O:[-] adegnrstu /P /Q /R /S[[+]n] /SX/T/U /UF /V /X/Z] [@file] source [+ ... [/A]/B] [TO:] target [...] [/A]/B]

source A file or list of files or a device to copy from
target A file, directory, or device to copy to
@file A text file containing the names of the source files, one per line (see [@file lists](#) for details)

/A(SCII) copy	/MD (Create target directory)
/A:... (Attribute select)	/N (Disable)
/B(inary copy)	/O(nly if no target)
/C(hanged source files)	/O:... (order)
/CF (changed 2s+ resolution)	/P(rompt)
/D (Copy encrypted files)	/Q(uiet)
/E (No error messages)	/R(eplace)
/F (No empty subdirectories)	/S(ubdirectories)
/FTP:A (ASCII copy)	/SX(single target directory)
/G (Display percentage)	/T(otals)
/H (Include hidden files)	/U(pdate target)
/I"text" (Match description)	/UF (update 2s+ resolution)
/J (Restartable)	/V(erify)
/K (Keep read-only attribute)	/W (one-way sync)
/L Copy symbolic links	/X(Clear archive)
/LD (create link)	/Y (suppress prompt)
/M(odified files)	/Z(overwrite)

See also: [ATTRIB](#), [MOVE](#), and [REN](#).

File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), [delayed variable expansion](#), and [include lists](#). Date, time, size or exclude ranges anywhere on the line apply to all source files. Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Internet

Can be used with [FTP / FTPS / TFTP / HTTP / HTTPS Servers](#).

Usage

If you don't specify any arguments, COPY will display its command dialog.

The simplest use of COPY is to make a copy of a file, like this example which makes a copy of a file called FILE1.ABC:

```
copy file1.abc file2.def
```

You can also copy a file to another drive and/or directory. The following command copies **FILE1** to the **\MYDIR** directory on drive **E**:

```
copy file1 e:\mydir
```

When you COPY files to or from an LFN drive, you must quote any file names which contain white space or special characters.

If you specify the /C, /CF, /R, /U, or /UF options, COPY will append a ! to the copy specifier if the target exists and is being overwritten. For example:

```
[d:] copy file1 file2  
file1 =>! file2
```

To emulate an approach used by some implementations of CMD, see the [COPYCMD](#) topic.

COPY sets three internal variables:

%_copy_dirs	The number of directories created
%_copy_files	The number of files copied
%_copy_errors	The number of errors

- **Copying Files**

You can copy several files at once by using [wildcards](#):

```
copy *.txt e:\mydir
```

You can also list several **source** files in one command. The following command copies 3 specific files from the current directory to the **\MYDIR** directory on drive **E**:

```
copy file1 file2 file3 e:\mydir
```

COPY also understands [include lists](#), so you can specify several different kinds of files in the same command. This command copies the **.TXT**, **.DOC**, and **.BAT** files from the **E: \MYDIR** directory to the root directory of drive **A**:

```
copy e:\mydir\*.txt;*.doc;*.bat a:\
```

If there is only one parameter on the line, COPY assumes it is the **source**, and uses the current drive and directory as the **destination**. For example, the following command copies all the **.DAT** files from the current directory on drive **A** to the current directory on the current drive:

```
copy a:*.dat
```

If there are two or more parameters on the line separated by spaces, then COPY assumes that the last parameter is the **destination** and copies all **source** files to this new location. If the **destination** is a drive, directory, or device name, the **source** files are copied individually to the new location. If the **destination** is a file name, the first **source** file is copied to the **destination**, and any additional **source** files are then appended to the new **destination** file.

For example, the first of these commands copies the **.DAT** files from the current directory on drive **A** individually to **C:\MYDIR** (which must already exist as a directory); the second appends all the **.DAT** files

together into one large file called C:\DATA (assuming C:\DATA is not a directory):

```
copy a:*.dat c:\mydir\  
copy a:*.dat c:\data
```

When you copy to a directory, if you add a backslash \ to the end of the name as shown in the first example above, COPY will display an error message if the name does not refer to an existing directory. You can use this feature to keep COPY from treating a mistyped **destination** directory name as a file name and attempting to append all your **source** files to a single **destination** file, when you really meant to copy them individually to a **destination** directory.

To copy text to or from the clipboard use CLIP: as the device name. Using CLIP: with non-text data will produce unpredictable results. See [Redirection](#) for more information on CLIP:.

- **Appending Files**

A plus sign + tells COPY to append two or more **source** files to a single **destination** file. If you list several **source** files separated with + and don't specify a **destination**, COPY will use the name of the first **source** file as the destination, and append each subsequent file to the first file.

For example, the following command will append the contents of MEMO2 and MEMO3 to MEMO1 and leave the combined contents in the file named MEMO1:

```
copy memo1+memo2+memo3
```

To append the same three files but store the result in BIGMEMO:

```
copy memo1+memo2+memo3 bigmemo
```

If no **destination** is specified, the destination file will always be created in the current directory even if the first **source** file is in another directory or on another drive. For example, this command will append C:\MEMMEMO2 and C:\MEMMEMO3 to D:\DATA\MEMO1, and leave the result in C:\MEMMEMO1:

```
[c:\mem] copy d:\data\memo1+memo2+memo3
```

You cannot append files to a device (such as a printer); if you try to do so, COPY will ignore the + signs and copy the files individually. If you attempt to append several **source** files to a **destination** directory or disk, COPY will append the files and place the copy in the new location with the same name as the first **source** file.

You cannot append a file to itself.

- **FTP Usage**

If you have appropriate permissions, you can copy to and from Internet URLs (FTP, TFTP and HTTP). Many FTP servers use case sensitive file systems. For example:

```
copy ftp://ftp.abc.com/xyz/index index
```

Files copied to or from FTP/HTTP Servers are normally transferred in binary mode. To perform an ASCII transfer use the [/L](#) switch. File descriptions are not copied when copying files to an Internet URL.

COPY supports the special syntax

```
copy con: ftp:...
```

to directly copy text from the console to an ftp location.

Wildcard characters such as * and ? will be treated as wildcards in FTP URLs, but will be treated as normal characters in HTTP URLs.

Note: The [/G](#) option (percentage copied) may report erratic values during transfer of files larger than 4 Gb (an ftp limitation) and during http downloads.

You can also use the [IFTP](#) command to start an FTP session on a server, and then use an abbreviated syntax to specify the files and directories you want. For more information, see [Using FTP/HTTP Servers](#) and [IFTP](#).

- **NTFS File Streams**

COPY supports file streams on NTFS drives. You can copy an individual stream by specifying the stream name, for example:

```
copy myfile:mystream stream.copy
```

If no stream name is specified the entire file is copied, including all streams. However, if you copy a file to a drive or device which does not support streams, only the file's primary data is copied; any additional streams are not processed.

See [NTFS File Streams](#) for additional details.

- **Advanced Features**

If your **destination** has wildcards in it, COPY will attempt to match them with the **source** names. For example, this command copies the .DAT files from drive **A** to C:\MYDIR and gives the new copies the extension .DX:

```
copy a:*.dat c:\mydir\*.dx
```

This feature can give you unexpected results if you use it with multiple **source** file names. For example, suppose that drive **A** contains XYZ.DAT and XYZ.TXT. The command:

```
copy a:\*.dat a:\*.txt c:\mydir\*.dx
```

will copy A:XYZ.DAT to C:\MYDIR\XYZ.DX. Then it will copy A:XYZ.TXT to C:\MYDIR\XYZ.DX, overwriting the first file it copied.

You can use [date, time, and size ranges](#) to further define the files that you want to copy. This example copies every file in the E:\MYDIR directory, which was created or modified yesterday, and which is also 10,000 bytes or smaller in size, to the root directory of drive **A**:

```
copy /[d-1] /[s0,10000] e:\mydir\* a:\
```

You can also use file exclusion ranges to restrict the list of files that would normally be selected with wildcards. This example copies every file in the E:\MYDIR directory except backup (.BAK or .BK) files:

```
copy /[!*.bak *.bk] e:\mydir\* a:\
```

COPY will normally process **source** files which do not have the hidden or system attribute, and will ignore the read-only and archive attributes. It will always set the archive attribute and clear the read-only attribute of **destination** files. In addition, if the **destination** is an existing file with the read-only attribute, COPY will generate an **Access Denied** error and refuse to overwrite the file. You can alter some of these behaviors with switches:

- /A:**.. Forces COPY to process **source** files with the attributes you specify after the :, or to process all **source** files regardless of attributes, if **/A:** is used by itself.
- /H** Forces COPY to process hidden and system **source** files, as well as normal files. The hidden and system attributes from each source file will be preserved when creating the **destination** files.
- /K** Retains the read-only attribute from each **source** file when creating the **destination** file. See **/K** below for a special note if you are running under Novell NetWare.
- /Z** Forces COPY to overwrite an existing **destination** file regardless of its attributes.

You can copy files to multiple destinations with the TO: option. For example, to copy **letter.doc** to three different directories:

```
copy letter.doc TO: \save\ f:\backups\ q:\letters\
```

Note: The wildcard expansion process will attempt to allow both CMD-style "extension" matching (assumes only one extension, at the end of the word) and the advanced **TCC** string matching (allowing things like ***.*.abc**) when an asterisk is encountered in the **destination** of a COPY command.

COPY supports [regular expression](#) back references in the target name. If you are using back references, you must also use a regular expression in the source name. The syntax is:

```
copy ::filename ::target
```

COPY supports connected web folders. If an HTML file (i.e., with an .htm or .html extension) is copied, COPY will look for a folder in the same directory with the same name and an extension of ".files". If it is found, the .files directory will be copied to the target directory. You can disable connected web folders by setting the registry key:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\NoFileFolderConnection=0
```

Options

The **/A** (ASCII copy) and **/B** (binary copy) options apply to the preceding filename and to all subsequent filenames on the command line until the file name preceding the next **/A** or **/B**, if any. All other options apply to all filenames on the command line, no matter where you put them.

Some options do not make sense in certain contexts, in which case COPY will ignore them. For example, you cannot prompt before replacing an existing file when the **destination** is a device such as the printer; there's no such thing as an "existing file" on the printer. If you use conflicting output options, like **/Q** and **/P**, COPY will generally take a "conservative" approach and give priority to the option which generates more prompts or more information.

- /= Display the COPY command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A If you use /A with a **source** filename, the file will be copied up to, but not including, the first Control-Z (ASCII: 26) character in the file. If you use /A with a **destination** filename, a Control-Z will be added to the end of the file. /A is the default when appending files, or when the **destination** is a device like **NUL**, rather than a disk file.

This option applies to the filename immediately preceding it, and to all subsequent filenames until the file name preceding the next /A or /B option.
- /A... Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. See the cautionary note under **Advanced Features** above before using /A: when both **source** and **destination** directories contain file descriptions. You must include the colon with this option to distinguish it from the /A switch, above. Do not use /A: with @file lists. See [@file lists](#) for details. Hidden or system files selected by this option overwrite hidden or system files.

You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /B If you use /B with a **source** filename, the entire file is copied; Ctrl-Z characters, if any, in the file are considered ordinary data to be copied. Using /B with a **destination** filename prevents addition of a Ctrl-Z to the end of the **destination** file. /B is the default unless source files are appended to the target file, or the target is a device, e.g., **NUL**.

This option applies to the filename immediately preceding it, and to all subsequent filenames until the file name preceding the next /A or /B option.
- /C Copy files only if the **destination** file exists and is older than the **source** (see also [/U](#)). This option is useful for updating the files in one directory from those in another without copying any files not already in the target directory. Before using /C in a network environment, be sure to read the note under [/U](#). Do not use /C with @file lists. See [@file lists](#) for details.
- /CF Copy files only if the **destination** file exists and is more than 2 seconds older than the **source** (see also [/C](#) and [/UF](#)). Do not use /CF with @file lists. See [@file lists](#) for details. (Not available in **TCC/LE**.)
- /D Force copy of an encrypted file even when the target will be decrypted (for CMD compatibility).
- /E (No error messages) Suppress all non-fatal error messages, such as **File not found** or **Can't copy file to itself**. Fatal error messages, such as **Drive not ready**, will still be displayed. This option is most useful in batch files and aliases.
- /F When used with /S, COPY will not create any empty subdirectories.
- /FTP:A Perform FTP transfers in ASCII mode, instead of the default binary mode.
- /G Displays the percentage copied, the transfer rate (in Kbytes/second), and the estimated time remaining. Useful when copying large files across a network or via FTP to ensure the copy is proceeding. When [/V](#) is also used, reports percentage verified.

- /H** Copy all matching files including those with the hidden and/or system attribute set. See the cautionary note under **Advanced Features** above before using /H when both **source** and **destination** directories contain file descriptions.
- /I"text"** (Match descriptions) Select **source** files by matching text in their descriptions. See [Description Ranges](#) for details.
- /J** Copy the file in restartable mode. The copy progress is tracked in the destination file in case the copy fails. The copy can be restarted by specifying the same source and destination file names. /J will not work with HTTP or FTP files.
- /K** (Keep read-only attribute) To maintain compatibility with CMD, COPY normally maintains the hidden and system attributes, sets the archive attribute, and removes the read-only attribute on the target file. /K tells COPY to also maintain the read-only attribute on the **destination** file. However, if the **destination** is on a Novell NetWare volume, this option will fail to maintain the read-only attribute. This is due to the way NetWare handles file attributes, and is not a problem in COPY.
- /L** (Windows Vista or later only) If the source is a symbolic link, copy the link to the target instead of the actual file.
- /LD** When used with /S, if the source is a symbolic or hard link to a directory, COPY will create the link in the target directory instead of copying the subdirectory tree.
- /M** Copy only those files with the archive attribute set, *i.e.*, those which have been modified since the last backup. The archive attribute of the **source** file will not be cleared after copying; to clear it use the /X switch, or use [ATTRIB](#). Do not use /M with [@file lists](#). See [@file lists](#) for details.
- /MD** Create the target directory if it doesn't exist. Note that you *must* either terminate the target directory name with a trailing \ or specify a filename component; otherwise COPY cannot tell what you want for the directory and what you want for the filename.
- /N** Do everything except actually perform the copy. This option is useful for testing what the result of a complex COPY command will be. /N displays how many files would be copied.

A /N with one or more of the following arguments has an alternate meaning:

- | | |
|----------|--|
| d | Skip hidden directories (when used with /S) |
| e | Don't display errors. |
| j | Skip junctions (when used with /S) |
| n | Don't copy/update the file descriptions |
| s | Don't display the summary. |
| t | Don't update the CD / CDD extended directory search database (<i>JPSTREE.IDX</i>). |
- /O** Only copy the source file if the target file doesn't exist.
- /O:...** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary

key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

The **/O:...** option saves all of the matching filenames and then performs the copy. This avoids the potential problem of copying files more than once.

- /P** Ask the user to confirm each *source* file. Your options at the prompt are explained in detail under [Page and File Prompts](#). Note: the [Copy Prompt on Overwrite](#) configuration option can be used to force prompting at the command line only. See also: the [**/Q**](#) option below.
- /Q** Don't display filenames, percentage copied, total number of files copied, etc. When used in combination with the [**/P**](#) option above, it will prompt for filenames but will not display the totals. This option is most often used in batch files. See also [**/T**](#).
- /R** Prompt the user before overwriting an existing file. Your options at the prompt are explained in detail under [Page and File Prompts](#). See also: the [Copy Prompt on Overwrite](#) configuration option. (For compatibility with CMD, a **/Y** option on the command line is changed to **/R**.)
- /S** Copy the subdirectory tree starting with the files in the **source** directory plus each subdirectory below that. The **destination** must be a directory; if it doesn't exist, COPY will attempt to create it. COPY will also attempt to create needed subdirectories on the tree below the **destination**, including empty **source** directories. If COPY /S creates one or more destination directories, they will be added automatically to the [extended directory search](#) database.

If you attempt to use COPY /S to copy a subdirectory tree into part of itself, COPY will detect the resulting infinite loop, display an error message and exit. Do not use **/S** with [@file lists](#) for details.
- If you specify a number after the **/S**, COPY will limit the subdirectory recursion to that number. For example, if you have a directory tree "**\a\b\c\d\e**", **/S2** will only affect the "a", "b", and "c" directories.
- If you specify a **+** followed by a number after the **/S**, COPY will not copy any files until it gets to that depth in the subdirectory tree. For example, if you have a directory tree **\a\b\c\d\e**, **/S+2** will not copy anything in **\a** or **\a\b**. (Not available in **TCC/LE**.)
- /SX** Copy the subdirectory tree to a single target directory (implies **/S**). (Not available in **TCC/LE**.) For example, to copy all of the .EXE files in "c:\files" and all of its subdirectories to the directory "d:\execfiles":

```
copy /sx c:\files\*.exe d:\exefiles\
```

- /T Turns off the display of filenames, like [/Q](#), but does display the total number of files copied.
- /U Copy each **source** file only if it is newer than a matching **destination** file or if a matching **destination** file does not exist (see also [/C](#)). This option is useful for keeping one directory matched with another with a minimum of copying. Do not use [/U](#) with @file lists. See [@file lists](#) for details. When used with file systems that have different time resolutions (such as FAT and NTFS), [/U](#) will attempt to use the "coarsest" resolution of the two.
- /UF Copy each **source** file only if it is more than 2 seconds newer than a matching **destination** file or if a matching **destination** file does not exist (see also [/CF](#) and [/U](#)). Do not use [/UF](#) with @file lists. See [@file lists](#) for details. (Not available in **TCC/LE**.)
- /V Verify each disk write by performing a true byte-by-byte comparison between the source and the newly-created target file. This option will significantly increase the time necessary to complete a COPY command. [/V](#) will not work for FTP, TFTP, or HTTP copies.
- /W Delete files in the target directory that don't exist in the source directory. (Use this instead of SYNC when you only want to synchronize "one way".)
- /X Clear the archive attribute from the source file after a successful copy. This option is most useful if you are using COPY to maintain a set of backup files. [/X](#) should not be used with multiple targets, because the archive attribute will be cleared after the first copy.
- /Y If you have the [COPY Prompt on Overwrite](#) option set, you can suppress the prompt with [/Y](#).
- /Z Overwrite **destination** files regardless of their attributes. Without this option, COPY will fail with an "Access denied error" if the **destination** file has its read-only attribute set, or (depending on other options) its hidden or system attribute set. Required to overwrite read-only targets regardless of other options. Required to overwrite hidden or system targets unless the source also has the attribute, and either [/H](#) or [/A:](#) is used to select it.

4.3.24 DATE

Purpose: Display and optionally change the system date

Format: DATE [/Fn /T /U] [mm -dd -yy]

mm	The month (1 - 12)
dd	The day (1 - 31)
yy	The year (80 - 99 or a 4- digit year)

/F(format)	/U (UTC date)
/T (Display only)	

See also: [TIME](#).

Usage:

If you simply type DATE without any parameters, you will see the current system date and time, and be

prompted for a new date. Press **Enter** if you don't wish to change the date. If you type a new date, it will become the current system date, which is included in the directory entry for each file as it is created or altered:

```
date
Thu Aug 18, 2011 9:30:06
Enter new date (mm-dd-yy) :
```

You can also enter a new system date by typing the DATE command plus the new date on the command line:

```
date 9-16-2011
```

You can use hyphens, slashes, or periods to separate the month, day, and year entries. The year can be entered as a 2-digit or 4-digit value. Two-digit years between 80 and 99 are interpreted as 1980 - 1999; values between 00 and 79 are interpreted as 2000 - 2079.

DATE adjusts the format it expects depending on your country settings. When entering the date, use the correct format for the country setting currently in effect on your system.

You can also use the international date format **yyyy-mm-dd**.

Option:

/F Date format to use (not available in **TCC/LC**). The formats are:

```
0 : Fri Jan 1, 2010
1 : 1/01/10
2 : Fri 1/01/2010
```

/T Displays the current date but does not prompt you for a new date. If a new date is specified in the same command as **/T** the new date will be ignored.

/U Display or enter the UTC date.

4.3.25 DATEMONITOR

Purpose: Monitor the current date and time

Format: DATEMONITOR [/C [yyyy-mm-dd hh:mm]]
DATEMONITOR yyyy-m-d hh:mm n command

n Number of repetitions (or **FOREVER**)

yyyy-mm-dd The date to match

hh:mm The time to match

command Command to execute when the specified date and time matches the current date and time

[/C\(lear\)](#)

Usage:

DATEMONITOR monitors the current date and time, and executes the specified command when the current date and time match the saved date and time. You can use a * in the date fields if you want to

run a command at a specific time every day. For example:

```
datemonitor *-* 23:59 forever echo It's almost midnight!
```

If you want to run a command on the first day of every month:

```
datemonitor *-*-1 00:01 forever echo It's the beginning of a new month!
```

If you don't enter any arguments, if DATEMONITOR is active it will display the repeat count and the command.

DATEMONITOR sets two environment variables when the date and time match and the trigger is set:

`_datemonitor` The current date in yyyy-mm-dd format

`_timemonitor` The current time in hh:mm (24-hour) format

The command line will be parsed and expanded before DATEMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. DATEMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

Options:

/C Remove date monitors. You can optionally specify a specific date monitor to remove by entering the date and time (which may include wildcards) for that monitor.

4.3.26 DEBUGMONITOR

Purpose: Monitor the OutputDebugString API

Format: DEBUGMONITOR [/C]
DEBUGMONITOR n command

n Number of repetitions (or **FOREVER**)

command Command to execute when condition is triggered

[/C\(lear\)](#)

Usage:

DEBUGMONITOR looks for any process calling the Windows OutputDebugString API.

DEBUGMONITOR will set the environment variable `_OUTPUTDEBUGSTRING` to the value specified in the OutputDebugString call.

The command line will be parsed and expanded before DEBUGMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. DEBUGMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

Options:

/C Remove the OutputDebugString monitor.

4.3.27 DEBUGSTRING

Purpose: Write text to the debugger for display

Format: DEBUGSTRING string.

Usage:

If the application has no debugger, the system debugger displays the message. If the application has no debugger and the system debugger is not active, DEBUGSTRING does nothing.

4.3.28 DEFER

Purpose: Execute a command after the batch file exits

Format: DEFER command

Usage:

A batch file can have multiple DEFER commands. They will be executed in first in, first out order when the batch file exits.

If you have variables on the DEFER command line, they will be expanded before the DEFER command is processed, not when **command** is executed. To delay variable expansion until **command** is executed, use single back quotes around the variable names, or double the %'s before the variable names.

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. DEFER will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

4.3.29 DEL / ERASE

Purpose: Erase one file, a group of files, or entire subdirectories

Format: DEL [ranges] [/A:[[-]+]rhsadecijopt /E /F /l"text" /K /L /N[defjnst] /O:[-]adegnrstu /P /Q /R /S[[+]n] /T /W[n] /X/Y /Z] [@file] file...

file The file, subdirectory, or list of files or subdirectories to erase.

@file A text file containing the names of the files to delete, one per line (see [@file lists](#) for details).

[/A: \(Attribute select\)](#)
[/B \(Delete after reboot\)](#)
[/E \(No error messages\)](#)
[/F\(orce delete\)](#)
[/I \(match descriptions\)](#)
[/K \(no Recycle Bin\)](#)
[/L \(delete symlinks\)](#)
[/N \(Disable\)](#)
[/O:... \(Order\)](#)

[/P\(rompt\)](#)
[/Q\(uiet\)](#)
[/R\(ecycle bin\)](#)
[/S\(ubdirectories\)](#)
[/T\(otal\)](#)
[/W\(ipe\)](#)
[/X\(remove empty subdirectories\)](#)
[/Y\(es to all prompts\)](#)
[/Z\(ap hidden and read-only files\)](#)

File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#). Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Internet

Can be used with [FTP/HTTP Servers](#).

Usage

DEL and ERASE are synonyms. You can use either one. In the description below, every reference to DEL applies equally to ERASE. If you don't specify any arguments, DEL will display its command dialog.

Use the DEL command with caution. The files and subdirectories that you erase may be impossible to recover without specialized utilities and a lot of work.

To erase a single file, simply enter the file name:

```
del letters.txt
```

You can also erase multiple files in a single command. For example, to erase all the files in the current directory with a `.BAK` or `.PRN` extension:

```
del *.bak *.prn
```

When you use DEL on an LFN drive, you must quote any file names which contain white space or special characters.

To exclude files from a DEL command, use a [file exclusion range](#). For example, to delete all files in the current directory except those whose extension is `.TXT`, use a command like this:

```
del /[!*.TXT] *
```

When using exclusion ranges or other more complex options you may want to use the `/N` switch first, to preview the effects of the DEL without actually deleting any files.

If you enter a subdirectory name, or a filename composed only of wildcards (* and/or ?), DEL asks for confirmation (Y or N) unless you specified the /Y option. If you respond with a Y, DEL will delete all the files in that subdirectory (hidden, system, and read-only files are only deleted if you use the /Z option). NOTE: The Windows command processor, CMD, behaves the same way but does not ask for confirmation if you use /Q to delete files quietly. If you want **TCC** to follow CMD's approach and skip the confirmation prompt when /Q is used, set the [Prompt on Wildcard Deletes](#) configuration option. Use caution if you disable this option, as this will allow DEL /Q to delete an entire directory without prompting for confirmation.

DEL displays the amount of disk space recovered, unless the /Q option is used (see below). It does so by comparing the amount of free disk space before and after the DEL command is executed. This amount may be incorrect if you are using a deletion tracking system which stores deleted files in a hidden directory, or if another program performs a file operation while the DEL command is executing.

Remember that DEL removes file descriptions along with files. Most deletion tracking systems will not be able to save or recover a file's description, even if they can save or recover the data in a file. This applies to the use of DEL with the Windows Recycle Bin, too - the description will be lost.

When a file is deleted without using the Recycle Bin, its disk space is returned to the operating system for use by other files. However, the contents of the file remain on the disk until they are overwritten by another file. If you wish to obliterate a file or wipe its contents clean, use the [/W](#) option, which overwrites the file before deleting it. Use this option with caution! Once a file is obliterated, it is impossible to recover. Remember: [/W](#) overrides using the Recycle Bin.

DEL returns a non-zero exit code if no files are deleted, or if another error occurs. You can test this exit code with the [%_?](#) internal variable, and use it with [conditional commands \(&& and ||\)](#).

Use caution when using wildcards with DEL on LFN drives, because **TCC**'s wildcard matching can match both short and long filenames. This can delete files you did not expect; see [LFN File Searches](#) for additional details.

DEL sets three internal variables:

%_del_dirs	The number of directories deleted
%_del_files	The number of files deleted
%_del_errors	The number of errors

- **Recycle Bin**

When you delete files with DEL, **TCC** does not move the deleted files to the Windows Recycle Bin by default. You can change this default with the [Delete to Recycle Bin](#) configuration option. If you have disabled the recycle bin, you can override the setting and place deleted files in the recycle bin with the [/R](#) option:

```
del /r letters.txt
```

If you have enabled Recycle Bin support, but want to override the default setting on a one-time basis, and delete some files without placing them in the recycle bin, use the [/K](#) option:

```
del /k letters.txt
```

You can also exclude files from the Recycle bin, even if [Delete to Recycle Bin](#) is enabled, or if the

command use the [/R](#) option, with the [RecycleExclude](#) environment variable.

- **FTP Usage**

If you have appropriate permissions, you can delete files on [FTP servers](#). For example:

```
del ftp://ftp.abc.com/index
```

You can also use the [IFTP](#) command to start an FTP session on a server and then use one of the following syntax examples:

```
del ftp:path/*.txt  
del ftp:/path/*.txt
```

The first syntax will normally be interpreted by the server as relative to the path you specified when you used the [IFTP](#) command to start the FTP session. The second syntax, with a slash before the path name, is interpreted as starting from the root.

- **NTFS File Streams**

DEL supports file streams on NTFS drives. You can delete an individual stream by specifying the stream name, for example:

```
del streamfile:s1
```

If no stream name is specified the entire file is deleted, including all streams.

See [NTFS File Streams](#) for additional details.

Options

- /=** Display the DEL command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.
- /A:** Delete only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow **/A:**. Do not use **/A:** with @file lists. See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /B** If DEL can't delete the file (for example, if access is denied) it will schedule it to be deleted at the next reboot.
- /E** Suppress all non-fatal error messages, such as "File Not Found." Fatal error messages, such as "Drive not ready," will still be displayed. This option is most useful in batch files and aliases.
- /F** This option has the same effect as [/Z](#) (see below): it deletes read-only, hidden, and system files as well as normal files.. It is included for compatibility with CMD.
- /I"text"** Select filenames by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow

the **/I** immediately, with no intervening spaces. You can select all filenames that have a description with **/I"[?]*"**, or all filenames that do not have a description with **/I"[]"**.

- /K** Physically delete files instead of sending them to the Windows Recycle Bin.
- /L** Delete symlinks instead of their contents. (Not available in **TCC/LE**.)
- /N** Do everything except actually delete the file(s). This is useful for testing the result of a DEL.

A **/N** with one or more of the following arguments has an alternate meaning:

- d** Skip hidden directories (when used with **/S**)
- e** Don't display errors
- f** Don't display the bytes freed in the summary
- j** Skip junctions (when used with **/S**)
- n** Don't update the file descriptions
- s** Don't display the summary
- t** Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*)

- /O:....** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

- /P** Prompt the user to confirm each erasure. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /Q** Don't display filenames as they are deleted, or the number of files deleted or bytes freed. If [Prompt on Wildcard Deletes](#) is disabled then **/Q** also disables the normal confirmation prompt when performing wildcard deletions (e.g. DEL *), for compatibility with CMD. Use caution if you disable [Prompt on Wildcard Deletes](#), as this will allow DEL /Q to delete an entire directory without prompting for confirmation. See also [/T](#).
- /R** Delete files to the Windows Recycle Bin.
- /S** Delete the specified files in this directory and all of its subdirectories. This is like a GLOBAL DEL, and can be used to delete all the files in a subdirectory tree or even a whole disk. Do not use **/S** with @file lists. See [@file lists](#) for details.

If you specify a number after the /S, DEL will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\etc", /S2 will only affect the "a", "b", and "c" directories.

If you specify a + followed by a number after the /S, DEL will not delete any files until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\etc, /S+2 will not delete anything in \a or \a\b. (Not available in **TCC/LE**.)

- /T** Don't display filenames as they are deleted, but display the total number of files deleted plus the amount of free disk space recovered.
- /W[n]** Overwrite the file contents using the DoD 5220.22-M (E) standard for secure deletion. (This overwrites every byte in the file with different values). You can optionally specify the number of passes (1-999); the default is 3. See also [DelWipePasses](#). Use this option to completely obliterate a file's contents from your disk. Once you have used this option it is impossible to recover the file even if you are using an undelete utility, because the contents of the file are destroyed before it is deleted. /W will override a [/R](#).
- /X** Removes empty subdirectories (only useful when used with [/S](#)). If DEL deletes one or more directories, they will be removed automatically from the [extended directory search database](#). DEL will display the directories being removed (with a trailing \).
- /Y** The reverse of [/P](#). It assumes a Y response to everything, including deleting an entire subdirectory tree. **TCC** normally prompts before deleting files when the name consists only of wildcards or a subdirectory name (see above); /Y overrides this protection and should be used with extreme caution!
- /Z** Delete read-only, hidden, and system files as well as normal files. Files with the read-only, hidden, or system attribute set are normally protected from deletion; /Z overrides this protection, and should be used with caution. Because [EXCEPT](#) works by hiding files, /Z will override an [EXCEPT](#) command. However, files specified in a [file exclusion range](#) will not be deleted by DEL /Z.

For example, to delete the entire subdirectory tree starting with *C:\UTIL*, including hidden and read-only files, without prompting (use this command with CAUTION!):

```
del /s /x /y /z c:\util\
```

4.3.30 DELAY

Purpose: Pause for a specified length of time

Format: `DELAY [/B /F /M time]
DELAY UNTIL [yyyy-mm-dd] hh:mm[:ss]`

time The number of seconds or milliseconds to delay.

[/B\(reak enabled\)](#)
[/F\(lush keyboard\)](#)

[/M\(illiseconds\)](#)

Usage:

DELAY is useful in batch file loops while waiting for something to occur. For example, to wait for 10 seconds:

```
delay 10
```

DELAY is most useful when you need to wait a specific amount of time for an external event, or check a system condition periodically. For example, this batch file checks the battery status (as reported by your Advanced Power Management drivers) every 15 seconds, and gives a warning when battery life falls below 30%:

```
do forever
    iff %_apmlife lt 30 then
        beep 440 4 880 4 440 4 880 4
        echo Low Battery! !
    endiff
    delay 15
enddo
```

The maximum **time** value is limited to about 585 million years in Windows. If you don't enter a **time**, the default is 1 second.

You can optionally wait until the specified date and time. If no date is specified, DELAY defaults to today.

TCC uses the minimum possible processor time during a DELAY, in order to allow other applications full use of system resources.

You can cancel a delay by pressing **Ctrl-C** or **Ctrl-Break**.

Options:

- /B** Allows terminating a DELAY by pressing a key.
- /F** Flush the keyboard buffer when DELAY ends. (Not available in **TCC/LE**.)
- /M** Count by milliseconds instead of seconds. Normally only used for delays of less than 1 second.

4.3.31 DESCRIBE

Purpose: Create, modify, or delete file and subdirectory descriptions

Format: Creating or modifying descriptions

```
DESCRIBE [ranges... /I"text"] [/A:atrlst /O:[-]adegnrstu] [@file] file [[/D]"description"] ...]
```

Description file updating

```
DESCRIBE /U [[d:\path\descript.ion] ...]]
```

file The file or files to operate on.

@file A text file containing the names of the files to describe, one per line (see [@file lists](#) for details).

"description" The description to attach to the file.

/A: (Attribute select)
 /D(description follows)
 /I (match description)

/O:... (Order)
 /U(pdate) descriptions file

See also: [@DESCRIPT](#), [DIR](#), and [SELECT](#)

File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#). Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Usage:

DESCRIBE adds descriptions to files and subdirectories. (Volume root directories cannot have descriptions.) The descriptions are displayed by [DIR](#) in single-column mode and by [SELECT](#), and can be retrieved using the [@DESCRIPT](#) function. Descriptions let you identify your files in much more meaningful ways than you can in a filename alone.

If you don't specify any arguments, DESCRIBE will display its command dialog.

You enter a description on the command line by typing the DESCRIBE, the filename, and the description in double quotes, like this:

```
describe memo.txt "Memo to Bob about party"
```

If you don't put a description on the command line, DESCRIBE will prompt you for it:

```
describe memo.txt
Describe "memo.txt" : Memo to Bob about party
```

If you use wildcards or multiple filenames with the DESCRIBE command and don't include the description text, you will be prompted to enter a description for each file. If you do include the description on the command line, all matching files will be given the same description.

When you use DESCRIBE on an LFN drive, you must quote **file** if it contains white space or special characters.

If you enter a quoted description on the command line, and the text matches the name of a file in the current directory, **TCC** will treat the string as a quoted file name, not as description text as you intended. To resolve this problem use the [/D](#) switch immediately prior to the quoted description (with no intervening spaces). For example, if the current directory contains the files *DATA.TST* and "*Test File*", the first of these commands will work as intended, but the second will not (in the second example the string "*test file*" will be treated as a second file name, when it is intended to be description text):

```
describe data.tst /D"test file"  correct command
describe data.tst "test file"    incorrect command
```

On LFN drives you will not see file descriptions in a normal [DIR](#) display, because [DIR](#) must leave space for the long filenames. To view the descriptions, use [DIR](#) /Z to display the directory in FAT format. See [DIR](#) for more details.

Each description can be up to 511 characters long. You can change this limit with the [Maximum Length](#) configuration option. In order to fit your descriptions on a single line in a standard DIR display, keep them

to 40 characters or less (longer descriptions are wrapped in the [DIR](#) output). DESCRIBE can edit descriptions longer than [Maximum Length](#) (up to a limit of 511 characters), but will not allow you to lengthen the existing text.

The descriptions are stored either in the NTFS SummaryInformation stream (if you have set the [NTFS Descriptions](#) configuration option), or in each directory in a hidden file called *DESCRIPT.ION*. Use the [ATTRIB](#) command to remove the hidden attribute from this file if you need to copy or delete it. *DESCRIPT.ION* is always created as a hidden file, but will not be rehidden by **TCC** if you remove the hidden attribute.

You can change the description file name with the [Description Filename](#) configuration option or the [SETDOS /D](#) command, and retrieve it with the [%_DNAME](#) internal variable. Use caution when changing the description file name, as changing the name from the default will make it difficult to transfer file descriptions to another system.

The description file is modified appropriately whenever you perform an internal command which affects it (such as [COPY](#), [MOVE](#), [DEL](#), or [RENAME](#)), but not if you use an external program (such as XCOPY or Explorer). You can disable description processing with the [Enable Descriptions](#) configuration option, or with [SETDOS /D](#).

When you [COPY](#), [MOVE](#) or [REN](#) files between two directories, both of which have descriptions, and you use switches which enable processing of hidden files (or you have removed the hidden attribute from *DESCRIPT.ION*), you must use caution to avoid overwriting existing file descriptions in the *destination* directory with the *DESCRIPT.ION* file from the *source* directory. See the notes under the **Advanced Features** sections of [COPY](#) and [MOVE](#) for additional details.

If you disable descriptions with the [SETDOS /D0](#) option, DESCRIBE will return with an error message.

Descriptions in Take Command

You can also describe directories and files with the List View in **Take Command** when it is in "Details" view. Double click on the "Description" column, enter or edit the description in the edit box, and press Enter to save.

Options:

- /= Display the DESCRIBE command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A: Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with @file lists. See [@file lists](#) for details.

You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

/D"description"

The quoted string following the /D switch without any separation is used as a description, not a file name, avoiding ambiguity in the meaning of quoted strings. See the **Usage** section above for details.

/I"text"

Select files by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the `/I` immediately, with no intervening spaces. You can select all filenames that have a description with `/I"[?]"`, or all filenames that do not have a description with `/I"[]"`. Do not use `/I` with @file lists. See [@file lists](#) for details.

/O:... Sort the files before processing. You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see [/T:acw](#)
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

/U Update the *DESCRIPT.ION* file (or the file specified by the [Description Filename](#) configuration option), deleting the entries for any nonexistent files. If no filename is supplied, DESCRIBE will process *DESCRIPT.ION* in the current directory. Otherwise, DESCRIBE will process *DESCRIPT.ION* in the specified path(s). This option may not be used in conjunction with other DESCRIBE options.

4.3.32 DESKTOP

Purpose: Create or switch desktops

Format: DESKTOP [/C [/N] newdesktopname]
DESKTOP desktopname

[/C\(reate\)](#) [/N\(o activate\)](#)

Usage:

DESKTOP will create a new Windows desktop, or switch to an existing desktop.

Option:

- /C** Create a new desktop.
- /N** Don't switch to the new desktop.

4.3.33 DETACH

Purpose: Start a console (character-mode) application in detached mode

Format: DETACH [/Q] command

[/Q\(uiet\)](#)

command The name of a command to execute, including an optional drive and path specification and any parameters. The name must be enclosed in double quotes if it contains any spaces.

See also: [START](#) and [TASKEND](#).

Usage:

When you start a program with DETACH, that program cannot use the keyboard, mouse, or video display. It is "detached" from the normal means of user input and output. However, you can redirect the program's standard I/O to other devices if necessary, using [redirection](#) symbols. In most cases, you should only DETACH text-mode programs, since most graphical applications cannot run without a screen or keyboard, or have their input and output redirected.

The **command** can be an internal command, external command, alias, or batch file. If it is not an external command, **TCC** will detach a copy of **TCC** to execute the command.

For example, the following command will detach a copy of **TCC** to run the batch file *XYZ.BTM*:

```
detach xyz.btm
```

You can also include any parameters or command line switches which the command knows how to interpret:

```
detach "xyz.btm Monday Nebraska"
```

If you prefer, you can use the Linux syntax of putting a trailing & on the command line instead of specifying DETACH. (**TCC** will convert it to a DETACH before executing the command.)

```
xyz.btm &
```

Once the program has started, **TCC** returns to the prompt immediately. It does not wait for a detached program to finish.

The Process ID of the detached program is returned in the [DETACHPID](#) variable.

You can use the [TASKEND](#) command to stop a detached program which does not terminate on its own.

Options:

/Q Don't display the new process's ID.

4.3.34 DIR

Purpose: Display information about files and subdirectories

Format: DIR [ranges] [options] [file...]

ranges one or more [ranges](#)

options one or more file selection or report format [options](#)

file The file, directory, or list of files or directories to display.

<u>/1</u>	1 column output	<u>/L</u>	Lower case
<u>/2</u>	2 column output	<u>/M</u>	suppress footer
<u>/4</u>	4 column output	<u>/N</u> [desfhjlmv]	New format or disable options
<u>/:</u>	show streams	<u>/O</u>	Order
<u>/A</u>	Attribute select	<u>/P</u>	Pause
<u>/B</u>	Bare (name only)	<u>/Q</u>	show owner
<u>/C</u>	show Compression	<u>/R</u>	disable wrap
<u>/D</u>	Disable color coding	<u>/S</u> n	show Subdirectories to depth <i>n</i>
<u>/E</u>	upper case	<u>/T</u>	show aTtribute
<u>/F</u>	Full path	<u>/T:</u>	time type
<u>/G[:n]</u>	allocated size	<u>/U</u>	show summary information
<u>/H</u>	Hide dots	<u>/V</u>	Vertical sort
<u>/I"text"</u>	description range	<u>/W</u>	Wide
<u>/J</u>	Justify names	<u>/X</u>	show short names
<u>/K</u>	suppress header	<u>/Z</u>	use FAT format

See also: [ATTRIB](#), [DESCRIBE](#), [PDIR](#), [SELECT](#), and [SETDOS](#).

File Selection

Supports [command dialog](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

Internet: Can be used with [FTP servers](#).

Usage:

DIR can be used to display information about files from one or more directories (local or remote), in a wide range of formats. Depending on the options chosen, you can display the file name, attributes, and size; the time and date of the last change to the file; the file description; and the file's compression ratio. You can also display information in 1, 2, 4, or 5+ columns, sort the files several different ways, use color to distinguish file types, and pause after each full screen.

If you want to produce customized output that will be subsequently parsed by another program or batch file, or if you need a special-purpose directory display, see the [PDIR](#) command. DIR and PDIR are related, but they do not have identical switches and they are not intended to produce identical output.

The various DIR displays are controlled through options or switches. The best way to learn how to use the many options available with the DIR command is to experiment. You will soon know which options you want to use regularly. You can select those options permanently by using the [ALIAS](#) command.

For example, to display all the files in the current directory, in 2 columns, sorted vertically (down one column then down the next), and with a pause at the end of each page:

```
dir /2/p/v
```

To set up this format as the default, using an alias:

```
alias dir=*dir /2/p/v
```

When you use DIR on an LFN drive, you must quote any file names which contain white space or special characters.

DIR sets three internal variables:

<code>%_dir_dirs</code>	The number of directories displayed
<code>%_dir_files</code>	The number of files displayed
<code>%_dir_errors</code>	The number of errors

The following sections group DIR's features together in several categories. Many of the sections move from a general discussion to more technical material. If you find some of the information in a category too detailed for your needs, feel free to skip to the beginning of the next section. The sections are:

- [Selecting Files](#)
- [Default DIR Output Format](#)
- [Switching Formats](#)
- [Multiple Column Displays](#)
- [Color-Coded Directories](#)
- [Redirected Output](#)
- [Other Notes](#)
- [Options](#)
- [FTP usage](#)

Selecting Files

DIR can display information about a single file or about several, dozens, hundreds, or thousands of files at once. To display information about a single file, just add the name of the file to the DIR command line:

```
dir january.wks
```

The simplest way to view information about several files at once is to use wildcards. DIR can work with the normal wildcard characters (*) and (?) and the [extended wildcards](#). For example to display all of the .WKS files in the current directory:

```
dir *.wks
```

To display all .TXT files whose names begin with A, B, or C:

```
dir [abc]*.txt
```

If you don't specify a filename, DIR defaults to * on LFN drives, and *.* on drives which do not support long file names. This default displays all non-hidden files and subdirectories in the current directory. If you specify a filename for a **non-LFN** drive which includes some wildcards, and does not include an extension, DIR will append .* to it to match all extensions.

If you link two or more filenames together with spaces, DIR will display all of the files that match the first name and then all of the files that match the second name. You may use a different drive and path for each filename. This example lists all of the .WKS and then all of the .WK1 files in the current directory:

```
dir *.wks *.wk1
```

If you use an [include list](#) to link multiple filenames, DIR will display the matching filenames in a single listing. Only the first filename in an include list can have a path; the other files must be in the same path. This example displays the same files as the previous example, but the .WKS and .WK1 files are intermixed:

```
dir *.wks;*.wk1
```

You can include files in the current or named directory plus all of its accessible subdirectories by using the **/s** option. This example displays all of the **.WKS** and **.WK1** files in the **D:\DATA** directory and each of its subdirectories:

```
dir /s d:\data\*.wks;*.wk1
```

You can also select files by their attributes by using the **/A** option. For example, this command displays the names of all of the subdirectories of the current directory:

```
dir /a:d
```

Finally, with the **/I** option, DIR can select files to display based on their descriptions (see [DESCRIBE](#) for more information on file descriptions). DIR will display a file if its description matches the text after the **/I** switch. The search is not case sensitive. You can use wildcards and extended wildcards as part of the text. For example, to display any file described as a "Test File" you can use this command:

```
dir /i"test file"
```

If you want to display files that include the words "test file" anywhere in their descriptions, use extended wild cards like this:

```
dir /i"*test file*"
```

To display only those files which do not have descriptions, use:

```
dir /I"[]"
```

In addition, you can use [ranges](#) to select or exclude specific sets of files. For example, to display all files modified in the last week, all files except those with a **.BAK** extension, and all files over 500 KB in size:

```
dir /[d-7]
dir /[!* .bak]
dir /[s500K]
```

You can mix any of these file selection techniques in whatever ways suit your needs.

Default DIR Output Format

DIR's output varies based on the type of volume or drive on which the files are stored. If the volume supports long file names, the default DIR format contains 4 columns: the date of the last file modification or write, the time of last write, the file size in bytes, and the file name. The name is displayed as it is stored on the disk, in upper, lower, or mixed case. DIR will wrap filenames from one line to the next if they are too long to fit the width of the display. The standard output format is:

```
Volume in drive C is unlabeled      Serial number is 3aaaf:c891
Directory of  C:\release\version12\*
2011-08-30  0:39          <DIR>    .
2011-08-30  0:39          <DIR>    ..
2011-08-25  11:30         <DIR>    tcmd
```

2011-08-25 23:07 4,801,840 tcmd.exe

(See **Switching Formats** below for information on changing the standard long filename format to allow room for file descriptions.)

On FAT volumes which do not support long file names, the default DIR format contains 5 columns: the file name, the file size in bytes, the date of the last write, the time of the last write, and the file's description. File names are listed in lower-case; directory names in upper case:

```
Volume in drive C is C - BOOTUP      Serial ...
Directory of C:\*
.
..
TEST        <DIR>      8-01-11  16:21
jpstree.idx   196967    8-28-11  17:57 JP fuzzy directory index
```

DIR's output is normally sorted by name, with directories listed first. You can change the sort order with the /O option. For example, these two commands sort the output by date. The first command lists the oldest file first; the second command lists the oldest file last:

```
dir /o:d
dir /o:-d
```

When displaying file descriptions, DIR wraps long lines to fit on the screen. DIR displays a maximum of 40 characters of text in each line of a description (unless your screen width allows a wider display). If you disable description wrapping with the /R option, the description is truncated at the right edge of the screen, and a right arrow is added at the end of the line to alert you to the existence of additional description text.

DIR's default output is sorted. It displays directory names first, with "<DIR>" inserted instead of a file size, and then filenames. DIR assumes that sequences of digits should be sorted numerically (for example, the file *DRAW2* is listed before *DRAW03* because 2 is numerically smaller than 03), rather than strictly alphabetically (where *DRAW2* would come second because "2" follows "0" in alphanumeric order). You can change the sort order with the /O option. When DIR displays file names in a multi-column format, it sorts file names horizontally unless you use the /V option to display vertically sorted output.

DIR's display can be modified in many ways to meet different needs. Most of the following sections describe the various ways you can change DIR's output format.

Switching Formats

On volumes which support long file names, you can force DIR to use a FAT-like format (file name first, followed by file information) with the /Z option. If necessary, DIR /Z truncates long file names on LFN drives, and adds a right arrow to show that the name contains additional characters.

The standard LFN output format does not provide enough space to show descriptions along with file names. Therefore, if you wish to view file descriptions as part of the DIR listing on a volume which supports long file names, you must use the /Z option.

DIR will display the alternate, short file names for files with long file names if you use the /X option. Used alone, /X causes DIR to display names in 2 columns after the size, time, and date: one column for

alternate or short file names and the other for long file names. If a file does not have a short or alternate name which is different from the long filename, the first filename column is empty.

If you use **/X** and **/Z** together, DIR will display the short or alternate file names in the FAT-style display format.

If you use the **/B** option, DIR displays just file names and omits the file size, time stamp, and description for each file, for example:

```
[c:\] dir w* /b
WINDOWS
WINNT
WINALIAS
WINENV.BTM
....
```

There are several ways to modify the display produced by **/B**. The **/F** option is similar to **/B**, but displays the full path and name of each file, instead of just its name. To view the same information for a directory and its subdirectories use **/B /S** or **/F /S**. You can use **/B /X** to display the short name of each file, with no additional information.

Multiple Column Displays

DIR has three options, **/2**, **/4**, and **/W**, that create multi-column displays.

The **/2** option creates a 2-column display. On drives which support long filenames, only the name of each file is displayed, with directory names placed in square brackets to distinguish them from file names. On drives which do not support long filenames, or when **/Z** or **/X** is used (see below), the display includes the short name, file size, and time stamp for each file.

The **/4** option is similar to **/2**, but displays directory information in 4 columns. On drives which do not support long filenames, or when **/Z** or **/X** is used (see below), the display shows the file name and the file size in kilobytes (KB) or megabytes (MB), with "<D>" in the size column for directories.

The **/W** option displays directory information in 5 or more columns, depending on your screen width. Each entry in a DIR /W display contains either the name of a file or the name of a directory. Directory names are placed in square brackets to distinguish them from file names.

If you use one of these options on a drive that supports long file names, and do not select an alternate display format with **/Z** or **/X**, the actual number of columns will be based on the longest name to be displayed and your screen width, and may be less than the number you requested (for example, you might see only three columns even though you used **/4**). If the longest name is too long to fit in on a single line the display will be reduced to one column, and each name will be wrapped, with "extra" blank lines added so that each name takes the same number of lines.

On LFN drives you can use **/Z** with any of the multi-column options to create a FAT-format display, with long names truncated to fit in the available space. If you use **/X**, the FAT-format display is also used, but short names are displayed (rather than truncated long names). The following table summarizes the effects of different options when using **TCC** on an LFN drive:

	default or /1	/2 or /4 columns	/W (wide)
Normal	date, time, size, LFN	2 - 4 columns, LFNs only	No. of columns based on longest LFN

/Z (FAT)	truncated LFN, size, date, time	2 - 4 columns, truncated LFN plus date, time, size	5+ columns, truncated LFNs only
/X (SFN)	date, time, size, SFN, LFN	2 - 4 columns, SFNs plus date, time, size	5+ columns, SFNs only
/X/Z	SFN, size, date, time	(Same as /X)	(Same as /X)

Color-Coded Directories

DIR can display each file name and the associated file information in a different color, depending on the file's extension.

To choose the display colors, you must either use the [SET](#) command to create an environment variable called [COLORDIR](#), or use the [Directory Colors](#) configuration option. If you use neither the variable nor the configuration option, DIR will use the default screen colors for all files.

If you use the COLORDIR variable, it will override the [Directory Colors](#) option. You may find it useful to use the COLORDIR variable for experimenting, then to set permanent directory colors with the [Directory Colors](#) option.

The format for both the COLORDIR environment variable and the [Directory Colors](#) option is:

```
ext ... :ColorName; ...
```

where "ext" is either a file extension (which may include wildcards) or one or more of the following file types:

type	files affected
ARCHIVE	Files with archive attribute set (modified since the last backup)
COMPRESSED	Compressed files
DIRS	Directories
ENCRYPTED	Encrypted files
HIDDEN	Hidden files
JUNCTION	Junctions or symbolic links
NORMAL	File with no attribute set
NOTINDEXED	Files whose content is not indexed
OFFLINE	Offline files
RDONLY	Read-only files
SPARSE	Sparse files
SYSTEM	System files
TEMPORARY	temporary files

and "ColorName" is any valid color name (see [Colors and Color Names](#) for information on color names).

Note that if a file uses one of the reserved file type names shown above as its extension (e.g. *xyz.hidden*) , that file will receive the color defined for the file type.

Unlike most color specifications, the background portion of the color name may be omitted for directory colors. If you don't specify a background color, DIR will use the current screen background color.

For example, to display .COM and .EXE files in red on the current background, .C and .ASM files in bright cyan on the current background, read-only files in green on white, and everything else in the

default color:

```
set colordir=com exe:red; c asm:bright cyan; rdonly:green on white
```

[Extended wildcards](#) can be used in directory color specifications. For example, to display *.BAK*, *.BAX*, and *.BAC* files in red, and everything else in the default color:

```
set colordir=BA[KXC]:red
```

You can combine attribute tests with the **.and.** / **.or.** / **.xor.** / **.not.** keywords. For example, to display directories that are also hidden in blue:

```
set colordir=dirs .and. hidden:blue
```

COLORDIR processes the line from left to right, and does not support parentheses.

Redirected Output

The output of the DIR command, like that of most other internal commands, can be [redirected](#) to a file, printer, serial port, or other device. However, you may need to take certain DIR options into account when you redirect DIR's output.

DIR wraps both long file names and file descriptions at the width of your display. Its redirected output will also wrap at the screen width. Use the **/R** option if you wish to disable wrapping of long descriptions.

If you redirect a color-coded directory to a file or a character device, DIR will remove the color data as it sends the directory information to a file.

To redirect DIR output to the clipboard, use **CLIP:** as the output device name, for example:

```
dir *.exe > clip:
```

FTP Usage

You can display directories on [FTP servers](#). For example:

```
dir ftp://ftp.microsoft.com/
```

You can also use the [IFTP](#) command to start an FTP session on a server, and then use a simplified syntax to specify the files and directories you want.

Other Notes

If you have selected a specific country code for your system, DIR will display the date in the format for that country. The default date format is U.S. (mm-dd-yy). The separator character in the file time will also be affected by the country code. Thousands and decimal separators in numeric displays are affected by the country code, and by the ThousandsChar and DecimalChar settings selected with the configuration dialogs or in the [.INI file](#).

DIR can generally display any file date between January 1, 1980 and December 31, 2099 if the date is supplied properly by the operating system.

If you are using NTFS disk compression, you can use the **/C** switch to view the amount of compression

achieved for each file. When you do, the compression ratio is displayed instead of the file's description. You can also sort the display by compression ratios with the **/O:c** switch. Details for both switches are in the Options section below. **/C** and **/O:c** will be ignored for uncompressed drives. **/C** will not display compression ratios on drives that support long file names unless you also use **/Z** to switch to the old-style short filename format.

If the OFFLINE attribute is set, DIR will display the file size enclosed in parentheses (for compatibility with CMD).

Options:

Options on the command line apply only to the filenames which follow the option, and options at the end of the line apply to the preceding filename only. This allows you to specify different options for different groups of files, yet retains compatibility with the traditional DIR command when a single filename is specified.

- /=** Display the DIR command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /1** Single column display -- display the filename, size, date, and time; also displays the description on drives which do not support long filenames. This is the default. If **/T** is used the attributes are displayed instead of the description; if **/C** or **/O:c** is used the compression ratio is displayed instead of the description. This option is most useful if you wish to override a default **/2**, **/4**, or **/W** setting stored in an alias. In Windows Vista or later on NTFS drives, single column displays will also show the target of symbolic links following the filename.
- /2** Two column display -- display just the name (on LFN drives), or display the filename, size, date, and time on other drives. See **Multiple Column Displays** above for more details.
- /4** Four column display -- display just the name (on LFN drives); or display the filename and size, in K (kilobytes) or M (megabytes) on other drives, with files between 1 and 9.9 megabytes in size displayed in tenths (*i.e.*, "2.4M"). See **Multiple Column Displays** above for more details.
- /:** Display file stream names and sizes on NTFS volumes. When combined with the **/B** or **/F** options, the size is omitted.

When **/:** is used in conjunction with **/B** (Bare), the file name is displayed on the first line, then any streams, indented two spaces, on subsequent lines:

```
c:\test\myfile.dat
xyz:$DATA
abc:$DATA
```

When **/:** is used in conjunction with **/F** (Full path), the file name is displayed on the first line, then any streams are appended to the filename on subsequent lines:

```
c:\test\myfile.dat
c:\test\myfile.dat:xyz
c:\test\myfile.dat:abc
```

- /A[:]** Display only those files that have the specified attribute(s) set. See [Attribute Switches](#) for

information on the attributes which can follow **/A:**.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

- /B** Suppress the header and summary lines, and display file or subdirectory names only, in a single column. This option is most useful when you want to redirect a list of names to a file or another program. If you use **/B** with **/S**, DIR will show the full path of each file (the same display as **/F**) instead of simply its name and extension. If you use **/B** with **/X** on an LFN drive, DIR will display the short name of each file instead of the long name. **/B** also sets **/H**.

/B1 will display relative paths when used with **/S**. (Normally, **/B** shows the full pathname for the file.)
- /C** Display per-file and total compression percentage on NTFS drives with compression enabled. **/C** only works in single-column mode; it is ignored if **/2**, **/4**, or **/W** is used.
- /D** Temporarily disable directory color coding. May be required when color-coded directories are used and DIR output is redirected to a character device like a serial port (e.g., COM1). **/D** is not required when DIR output is redirected to a file.
- /E** Display filenames in upper case.
- /F** Display each filename with its drive letter and path in a single column, without other information. If you use **/F** with **/X**, the "short" version of the entire path is displayed.
- /G[:n]** Display the allocated disk space instead of the actual size of each file. You can optionally specify the disk cluster size to be used by **/G**. (DIR will normally query the system for the cluster size on the specified drive, but you can override with **/G:n** if you know that the returned info is incorrect, or if you want to find the size required if the specified files were moved to another device with a different cluster size.)
- /H** Suppress the display of the **".** and **..** directories.
- /I"text"** Select filenames by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the **/I** immediately, with no intervening spaces. You can select all filenames that have a description with **/I"[?]*"**, or all filenames that do not have a description with **/I"[]"**.

The **/I** option may be used to select files even if descriptions are not displayed (for example, if **/2** is used). However, **/I** will be ignored if **/C** or **/O:c** is used.
- /J** Justify (align) filename extensions and display them in the FAT format. If on an LFN drive, you must also specify the **/X** and **/Z** options.
- /K** Suppress the header (disk and directory name) display.
- /L** Display file and directory names in lower case.
- /M** Suppress the footer (file and byte count totals) display.
- /N** Use the long filename display format, even if the files are stored on a volume which does not support long filenames. See also **/Z**.

A /N with one of the following arguments has an alternate meaning:

- d** Skip hidden directories (when used with /S)
- e** Don't display an error message if no files match.
- f** Don't display "bytes free" in the summary
- h** Don't display the header
- j** Skip junctions (when used with /S)
- l** Don't display the link name for symbolic links
- m:n** Display a maximum of *n* directory entries
- s** Don't display the summary.
- v** Don't display the volume information.

- /O** Set the sorting order. You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:
- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
 - Reverse the sort order for the next sort key
 - a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
 - c** Sort by compression ratio (the least compressed file in the list will be displayed first). For single-column directory displays in the short filename format, the compression ratios will be used as the basis of the sort and will also be displayed. For wider displays (**/2**, **/4**, and **/W**) and displays in LFN format, the compression ratios will be used to determine the order but will not be displayed. For information on supported compression systems see **/C** above.
 - d** Sort by date and time (oldest first); also see **/T:acw**
 - e** Sort by extension
 - g** Group subdirectories first, then files
 - i** Sort by file description (ignored if **/C** or **/O:c** is also used)
 - o** Sort by owner
 - r** Reverse the sort order for all options
 - s** Sort by size
 - t** Same as **d**
 - u** Unsorted
- /P** Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /Q** Display the file or directory owner (NTFS and remote directories only).
- /R** Forces long descriptions to be displayed on a single line, rather than wrapped onto two or more lines. Use **/R** when output is redirected to a character device, such as a serial port or the printer; or when you want descriptions truncated, rather than wrapped, in the on-screen display.
- /S** Display file information from the current directory and all of its accessible subdirectories. DIR will only display headers and summaries for those directories which contain files that match the filename(s), ranges, and attributes that you specify on the command line. DIR will display hidden subdirectories for compatibility with CMD.

If you specify a number after the /S, DIR will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\e", /S2 will only affect the "a", "b", and "c" directories.

If you specify a + followed by a number after the /S, DIR will not display any filenames until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\e, /S+2 will not display the contents of \a or \a\b. (Not available in **TCC/LE**.)

- /T** Display the filenames and attributes in the format **RHSADENTPCOIJ**, regardless of volume type:

R	Read-only	A	Archive
H	Hidden	D	Subdirectory
S	System	C	Compressed
E	Encrypted	O	Offline
N	Normal	I	Not content-indexed
T	Temporary	L	Junction or symbolic link
P	Sparse file		

Attributes which are set are represented by their letter, unset attributes by the _ (underscore) character.

If you wish to add another option after **/T**, you must start the next option with a forward slash. If you don't, **TCC** will interpret the **/T** as the **/T:{acw}** time display switch (see below) and the following character as a time selector. For example:

dir /tz	incorrect, will display an error
dir /t/z	correct

- /T:a|c|w[u]** Specify which of the date and time fields on a drive which supports long filenames should be displayed and used for sorting:

- a** Last access date and time (on VFAT volumes access time is always midnight).
- c** Creation date and time.
- w** Last modification (write) date and time (default).

If you append a **u** after the field, DIR will display the file time in UTC.

- /U** Only display the number of files, the total file size, and the total amount of disk space used. Information on individual files is not displayed. **/U1** will display summaries for each directory, but no total summary for each parent directory. **/U2** displays the grand total only.
- /V** Display the filenames sorted vertically rather than horizontally (use with the [/2](#), [/4](#) or [/W](#) options).
- /W** Display filenames only, horizontally across the screen. On drives which do not support long filenames, or when used with **/Z** or **/X**, **/W** displays as many columns as it can fit into **TCC** window, using 16 characters in each column. Otherwise (*i.e.*, when long filenames are displayed) the number of columns depends on the width of the longest name in the listing. See **Multiple Column Displays** above for more details.
- /X** Display both the short name (8-character name plus 3-character extension) and the long name of each file on an LFN drive. In normal single-column output the short name is

displayed first, followed by the long name. The short name column is left blank if the short name and long name are the same. On **NTFS** volumes this means case insensitive match, but on **VFAT** volumes this means case sensitive match (i.e., no lower case letters in the **SFN**). **/X** also selects short filenames in the **/2**, **/4**, **/B**, **/W**, and **/Z** displays, and short file and path names in the **/F** display.

- /Z** Display filenames on LFN drives in the old-style format, with the filename on the left and the description (when available) on the right. Long names will be truncated to 12 characters unless **/X** is also used. If the name is longer than 12 characters, it will be followed by a → "right arrow" symbol to show that one or more characters have been truncated. If a [description file](#) exists, **/Z** defaults to using the name of the . and .. directories as description for those entries

4.3.35 DIRHISTORY

Purpose: Display, add to, clear, or read the directory history list

Format: DIRHISTORY [/A directory /F /G /L /N /P /R filename /Tn]

directory	The name of a directory to be added to the directory history.
filename	The name of a file containing entries to be added to the directory history.

/A(dd)	/N(o duplicates)
/F(ree)	/P(ause)
/G(labal)	/R(ead)
/L(ocal)	/T (display last n lines)

See also: [HISTORY](#).

File Selection

Supports [command dialog](#).

Usage

Every time you change to a new directory or drive, **TCC** saves the previous directory in an internal directory history list. The [directory history window](#) allows you to use the list to return to a previous directory. See also: [directory navigation](#).

The DIRHISTORY command lets you view and manipulate the directory history list directly. If no parameters are entered, DIRHISTORY will display the current directory history list:

```
dirhistory
```

With the options explained below, you can clear the list, add new directories to the list without changing to them, save the list in a file, or read a new list from a file.

The number of directories saved in the directory history list depends on the length of each directory name. The list size can be specified at startup with the [Directory History Buffer Size](#) configuration option. The default size is 4,096 characters.

Your directory history list can be stored either locally (a separate history list for each copy of **TCC**) or globally (all copies of **TCC** share the same list). For details see the discussion of [local and global](#)

[history lists](#). If you use global lists, [SHRALIAS](#) can save the list when no copy of **TCC** is active, as long as you do not restart Windows.

You can save the directory history list by redirecting the output of DIRHISTORY to a file. This example saves the history to a file called *DIRHIST* and reads it back again.

```
dirhistory > dirhist
.....
dirhistory /r dirhist
```

Because the directory history stores each name only once, you don't have to delete its contents before reading back the file unless you want to delete the directories that were visited by the intervening commands.

TCC can also load and save the history list automatically if you use the [Directory History File](#) configuration option.

Options

- /=** Display the DIRHISTORY command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A** Add a directory to the directory history list.
- /F** Erase all entries in the directory history list.
- /G** Switch from a local to a global directory history list.
- /L** Switch from a global to a local directory history list.
- /N** Removes duplicate entries (oldest first) from the directory history list.
- /P** Wait for a key after displaying each page of the list. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /R** Read the directory history from the specified file and append it to the list currently held in memory.
- /Tn** Display the last *n* lines of the directory history. If *n* is negative, skip the first *-n* lines of the directory history. (Not available in **TCC/LE**.)

4.3.36 DIRS

Purpose: Display the current directory stack

Format: DIRS [+n -n /Q]

+n / -n Rotate the directory stack up or down *n* entries
[/Q\(quiet\)](#)

See also: [PUSHD](#), [POPD](#), [@DIRSTACK](#) and [Directory Navigation](#).

Usage:

The [PUSHD](#) command adds the current default drive and directory to the directory stack, a list maintained by **TCC**. The [POPD](#) command removes the top entry of the directory stack and makes that drive and directory the new default. The DIRS command displays the contents of the directory stack, with the most recent entries last (i.e., the next POPD will retrieve the last entry that DIRS displays).

For example, to change directories and then display the directory stack:

```
[c:\] pushd c:\database
[c:\database] pushd d:\wordp\memos
[d:\wordp\memos] dirs
c\
c:\database
```

The directory stack holds 4096 characters, enough for 80+ typical drive and directory entries.

Options

/Q Don't display the directory stack (only useful when combined with +n or -n).

4.3.37 DISKMONITOR

Purpose: Monitor free disk space

Format: DISKMONITOR [/C [disk]]
DISKMONITOR disk size command

disk	Disk drive to monitor
size	Minimum free disk space
command	Command to execute when condition is triggered

[/C\(lear\)](#)

Usage:

If the free disk space for the drive drops below the specified size, DISKMONITOR will execute the specified command. For example, to send an email when the C: drive has less than 2Gb free:

DISKMONITOR C: 2Gb sendmail bob@bob.com "Disk Status" Drive C: is full!

The drive can also be a sharename. The size format is the same as that used for size ranges (i.e., either a number or a number with an appended k, K, m, M, g, G, t, or T).

The command line will be parsed and expanded before DISKMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. DISKMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

If you don't enter any arguments, DISKMONITOR will display the disk drives it is currently monitoring.

DISKMONITOR will poll the drives it is monitoring once every 10 seconds.

Options:

- /C If *disk* is specified, remove the monitor for that disk drive. Otherwise, remove all disk monitors.

4.3.38 DO

Purpose: Create loops in batch files

Format: DO loop_control
 commands
 [\[ITERATE\]](#)
 commands
 [\[LEAVE \[n\]\]](#)
 commands
ENDDO

Loop_control formats

```
DO count
DO FOREVER
DO varname = start TO end [BY step] [(command)]
DO WHILE condition [(command)]
DO UNTIL condition [(command)]
DO UNTIL DATETIME date time [(command)]
DO FOR n [SECONDS | MINUTES | HOURS] [(command)]
DO varname IN [range...] /D"directory" [/I:"text" /S[[-+]n] /A:[-+]+]rhsadecijopt /O:[-]adegnrstu
fileset [(command)]
DO varname IN [/T"delimiters"] /L stringset [(command)]
DO varname IN /C stringset [(command)]
DO varname IN /L stringset [(command)]
DO varname in /P command ... [(command)]
DO varname IN /Q stringset [(command)]
DO varname IN @file [(command)]
```

count	Integer in the range [0, 2147483647], or an internal variable or variable function that evaluates to such a value, specifying the number of times the loop is executed.
varname	The environment variable containing the current value of the loop index, or the current filename or string, or the current line from a file. Do not prefix the variable name with %.
start, end, step	Integers in the range [-2147483647, 2147483647] or internal variables or variable functions that evaluate to such values, controlling the number of times the loop is executed.
condition	A conditional expression to determine whether or not the loop should be executed
fileset	A filename or list of filenames, possibly using wildcards
stringset	An arbitrary set of strings. Wildcards are not interpreted.
file	A file each line of which contains a string the loop is to be executed for
range	A date, time, size or exclusion range. At most one of each, in any order.
commands	One or more commands to execute each time through the loop. If you use

multiple commands, they must be separated by command separators or be placed on separate lines.

date	The loop termination date in ISO 8601 format
time	The loop termination time in 24-h hh:mm:ss format
/A:	Attribute select
/C	Loop through each character in expression
/D"directory"	Start directory
/I"text"	(Match description) Description range.
/L(iteral)	Members of set are strings, not filenames
/O:... (Order)	Sort order
/P	Parse the output of the command, saving the next output line in varname each time the loop is executed.
/Q	Like /L , but treats double quoted arguments (with embedded whitespace) as a single argument.
/S	Perform the loop in the current directory and all its subdirectories

Supports extended [wildcards](#), [ranges](#), and [include lists](#) for the **set**. Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Usage

DO can be used in [batch files](#), aliases, or at the command prompt. To use them in aliases or at the prompt, you need to define the DO on a single line, and enclose the body of the DO loop in a command group following the DO expression. (There is no ENDDO statement in a single-line DO). For example:

```
do count=1 to 10 by 1 (echo count=%count)
```

If the last argument on the line is a single **,** it is interpreted as the beginning of a command group. DO will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing **.**

DO sets four internal variables:

%_do_dirs	The number of directories traversed (with /S)
%_do_files	The number of directory entries (files or subdirectories) processed
%_do_errors	The number of errors
%_do_loop	The number of times the DO loop has been executed

Types of DO Loops

DO can be used to create several different kinds of loops.

- DO **count**, is a counted loop. The batch file lines between DO and ENDDO are repeated **count** times. TCC does not provide the user with the count of how many times the loop has been executed, though it is possible for the user to create a such a mechanism. For example::

```
set ct=0
do 5
  beep
  set ct=%@inc[%ct]
enddo
```

- DO FOREVER creates an endless loop. You must use [LEAVE](#) or [GOTO](#) to exit such a loop.

- DO **varname = start TO end [BY step]** is similar to a "for loop" in programming languages like BASIC. DO creates an environment variable, **varname**, and sets it equal to the value **start**. If **varname** already exists in the environment, it will be overwritten. DO then begins the loop process by comparing the value of **varname** with the value of **end**. If **step** is positive or not specified, and **varname** is less than or equal to **end**, DO executes the batch file lines up to the ENDDO. Next, DO adds to the value of **varname** either the value of **step** if BY **step** is specified, or 1, and repeats the compare and execute process until **varname** is greater than **end**. This example displays the even numbers from 2 through 20:

```
do i = 2 to 20 by 2
echo %i
enddo
```

DO can also count down, rather than up. If **step** is negative, **varname** will be decreased by the absolute value of **step** with each loop, and the loop will stop when **varname** is less than **end**. For example, to display the even numbers from 2 through 20 in reverse order, replace the first line of the example above with:

```
do i = 20 to 2 by -2
```

- DO WHILE **condition** evaluates **condition** each time through the loop as a [conditional expression before](#) executing the loop, and will execute it only if it is true. If **condition** is FALSE when the DO is first executed, the loop will never be executed.
- DO UNTIL **condition** evaluates **condition** as a [conditional expression after](#) execution of the loop, and repeats the loop only if it is FALSE. Therefore, the statements within the loop will always be executed at least once.
- DO UNTIL DATETIME **date time** executes the loop until the current date and time is equal to or greater than the specified date (ISO format) and time (24-hour format). The date and time can be in either YYYY-MM-DD HH:MM:SS or YYYYMMDDHHMMSS format. (The date and/or time can be a variable.)
- DO FOR **n** SECONDS | MINUTES | HOURS executes the loop for the specified amount of time.
- DO **varname IN filespec** executes the commands between DO and ENDDO by creating an environment variable, **varname**, and setting it equal to every filename in the **filespec**, ignoring items not matching file or directory names. This is similar to the **set** used in the [FOR](#) command, but it can only include file and directory names, not arbitrary text strings. If **varname** already exists in the environment, it will be overwritten (unlike the control variable in [FOR](#)). For example:

```
do x in *.txt
...
enddo
```

will execute the loop once for every `.TXT` file in the current directory; each time through the loop the variable `x` will be set to the name of the next file that matches the file specification. The order of matches is dependent on the file system, and is totally unrelated to any characteristics of the filenames matched.

If, between DO and ENDDO, you create a new file that could be included in the list of files, it may or may not appear in an iteration of the DO loop. Whether the new file appears depends on its physical

location in the directory structure, a condition over which **TCC** has no control.

To use date, time, size, description, or file exclusion [ranges](#) for the **set** place them just before the filename(s), for example:

```
do x in /[d9-1-2011,9-31-2011] *.txt
```

- DO **varname** IN /L **stringset** executes the commands between DO and ENDDO once for every string literal in **stringset**, setting **varname** to each in turn.
- DO **varname** IN /C **stringset** executes the commands between DO and ENDDO once for every character in **stringset** (including whitespace and special characters), setting **varname** to each in turn.
- DO **varname** IN @**file** executes the commands between DO and ENDDO once for every line in **file**, setting **varname** to the content of each one in turn. Beware of characters with special meaning to **TCC**, such as redirection and piping symbols, within the file (use [SETDOS](#) /X as needed).

To execute the loop once for each line of text in the clipboard, use **CLIP:** as the file name (e.g. DO X IN @CLIP:). **CLIP:** will not return any data unless the clipboard contains text. See [Redirection](#) for more information on **CLIP:**

Special DO keywords: ITERATE and LEAVE

Two special keywords, ITERATE and LEAVE, may be used inside a DO / ENDDO loop. ITERATE ignores the remaining commands inside the loop and returns to the beginning of loop for another iteration, unless DO determines that the loop is finished. LEAVE exits from the current DO loop and continues with the command following its ENDDO. Both keywords may be repeated as often as desired. Both ITERATE and LEAVE are most often used in an [IF](#) or [IFF](#) command (group):

```
do while "%var" != "%val1"
  ...
  if "%var" == "%val2" leave
enddo
```

LEAVE accepts an optional numeric argument (≥ 1) which specifies the DO nesting level you want to leave. For example, "LEAVE 2" will exit two nested DO loops. (Not available in **TCC/LE**.)

Usage Notes

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

DO loops can be nested, i.e. you can have a DO / ENDDO loop within another DO / ENDDO loop.

You can exit from all DO / ENDDO loops in a batch file by using [GOTO](#) to a line past the corresponding ENDDO. However, be sure to read the cautionary notes about [GOTO](#) and DO under the [GOTO](#) command before using GOTO in any other way inside any DO loop.

You cannot use [RETURN](#) to return from a [GOSUB](#) while inside a DO loop.

Note: Do not confuse the DO command with the unrelated optional **do** keyword of the [FOR](#) command.

Options:

/A: Select the files in a [DO](#) x IN . . . by their specified attribute(s). See [Attribute Switches](#) for information on the attributes which can follow **/A:**.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

/C For each loop, assign the next character (including whitespace and special characters) in the expression to the DO variable.

/D"directory" Set the start directory (for use with **/S**).

/I"text" Select files in a [DO](#) x IN ... by matching **text** in their descriptions. See [Description Ranges](#) for details.

/L The parameters following [DO](#) x IN **/L** are strings, not filenames. Each parameter will be assigned in sequence, from left to right, to the loop control variable on consecutive passes through the loop. **/L** will not treat double quotes as delimiters; use **/Q** if you want to pass arguments with embedded white space.

/N Disable options:

- d** Skip hidden directories (when used with **/S**)
- j** Skip junctions (when used with **/S**)

/O:... Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

The **/O:...** option saves all of the matching filenames and then performs the requested operation. This avoids the potential problem of processing files more than once. (Not available in **TCC/LE**.)

/P For each loop, assign the next output line from **command** to the DO variable. (Not available in **TCC/LE**.)

/Q The parameters following [DO](#) x IN **/Q** are strings, not filenames. Each parameter will be assigned in sequence, from left to right, to the loop control variable on consecutive passes

through the loop. Unlike /L, /Q will treat double quoted arguments with embedded whitespace as a single argument.

- /S** Perform the DO loop in the current directory and then on all of its subdirectories. (DO also supports /R as a synonym, for compatibility with FOR.)

If you specify a number after the /S, DO will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\e", /S2 will only affect the "a", "b", and "c" directories.

If you specify a + followed by a number after the /S, DO will not execute **command** until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\e, /S+2 will not execute **command** in \a or \a\b. (Not available in **TCC/LE**.)

- /T"text"** Specify the delimiters to be used when parsing a string set.

4.3.39 DRAWBOX

Purpose: Draw a box on the screen

Format: DRAWBOX ulrow ulcol lrow lcol style [BRight] fg ON [BRight] bg [FILL [BRight] bgfill] [ZOOM] [SHadow]

ulrow	Row for upper left corner
ulcol	Column for upper left corner
lrow	Row for lower right corner
lcol	Column for lower right corner
style	Box drawing style:
	0 No lines (box is drawn with blanks)
	1 Single line
	2 Double line
	3 Single line on top and bottom, double on sides
	4 Double line on top and bottom, single on sides
fg	Foreground character color
bg	Background character color
bgfill	Background fill color (for the inside of the box)

See also: [DRAWHLINE](#) and [DRAWVLINE](#).

Usage:

DRAWBOX is useful for creating attractive screen displays in batch files.

For example, to draw a box around the edge of an 80x25 window with bright white lines on a blue background:

```
drawbox 0 0 24 79 1 bri whi on blu fill blu
```

See [Colors and Color Names](#) for details about colors.

If you use ZOOM, the box appears to grow in steps to its final size. The speed of the zoom operation depends on the speed of your computer and video system.

If you use SHADOW, a drop shadow is created by changing the characters in the row under the box and the 2 columns to the right of the box to normal intensity text with a black background (this will make characters displayed in black disappear entirely).

The row and column values are zero-based, so on a standard 25 line by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits). DRAWBOX checks for valid row and column values, and displays a "Usage" error message if any values are out of range.

The maximum **row** value is determined by the current height of the **TCC** window. The maximum **column** value is determined by the current virtual screen width (see [Resizing the Take Command Window](#) for more information).

If **ulrow** is set to 999, **lrow** is assumed to be the desired height, and the box will be centered vertically. If **ulcol** is set to 999, **lcol** is assumed to be the desired width, and the box will be centered horizontally.

Unlike DRAWHLINE and DRAWVLINE, DRAWBOX does not automatically connect boxes to existing lines on the screen with the proper connector characters. If you want to draw lines inside a box and have the proper connectors drawn automatically, draw the box first, then use DRAWHLINE and DRAWVLINE to draw the lines.

DRAWBOX uses the standard line and box drawing characters in a Unicode or U.S. English extended ASCII character set. If you use an ASCII or raster font which does not include these line drawing characters, the box or lines will not be displayed correctly.

4.3.40 DRAWHLINE

Purpose: Draw a horizontal line on the screen

Format: DRAWHLINE *row column len style [BRlight] fg ON [BRlight] bg*

row	Starting row
column	Starting column
len	Length of line
style	Line drawing style: 1 Single line 2 Double line
fg	Foreground character color
bg	Background character color

See also: [DRAWBOX](#) and [DRAWVLINE](#).

Usage:

DRAWHLINE is useful for creating attractive screen displays in batch files. It detects other lines and boxes on the display, and creates the appropriate connector characters when possible (not all types of lines can be connected with the available characters).

For example, the following command draws a double line along the top row of the display with green characters on a blue background:

```
drawhline 0 0 80 2 green on blue
```

The *row* and *column* values are zero-based, so on a 25 line by 80 column display, valid *rows* are 0 - 24 and valid *columns* are 0 - 79.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits). If either value is out of range, DRAWHLINE displays a "Usage" error message.

The maximum *row* value is determined by the current height of the **TCC** window. The maximum *column* value is determined by the current virtual screen width (see [Resizing the Take Command Window](#) for more information).

If *row* is set to 999, the line will be centered vertically. If *column* is set to 999, the line will be centered horizontally.

See [Colors and Color Names](#) for details about colors.

DRAWHLINE uses the standard line and box drawing characters in a Unicode or U.S. English extended ASCII character set. If you use an ASCII or raster font which does not include these line drawing characters, the box or lines will not be displayed correctly.

4.3.41 DRAWVLINE

Purpose: Draw a vertical line on the screen

Format: DRAWVLINE *row* *column* *len* *style* [BRIght] *fg* ON [BRIght] *bg*

<i>row</i>	Starting row
<i>column</i>	Starting column
<i>len</i>	Length of line
<i>style</i>	Line drawing style:
	1 Single line
	2 Double line
<i>fg</i>	Foreground character color
<i>bg</i>	Background character color

See also: [DRAWBOX](#) and [DRAWHLINE](#).

Usage:

DRAWVLINE is useful for creating attractive screen displays in batch files. It detects other lines and boxes on the display, and creates the appropriate connector characters when possible (not all types of lines can be connected with the available characters).

For example, to draw a double width line along the left margin of the display with bright red characters on a black background:

```
drawvline 0 0 25 2 bright red on black
```

The *row* and *column* values are zero-based, so on a 25 line by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79. Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits). If either value is out of range, DRAWVLINE displays a "Usage" error message.

The maximum *row* value is determined by the current height of the **TCC** window. The maximum *column* value is determined by the current virtual screen width (see [Resizing the Take Command Window](#) for

more information).

See [Colors and Color Names](#) for details about colors.

DRAWVLINE uses the standard line and box drawing characters in a Unicode or U.S. English extended ASCII character set. If you use an ASCII or raster font which does not include these line drawing characters, the box or lines will not be displayed correctly.

4.3.42 ECHO

Purpose: Enable or disable batch file or command line echoing, display the echoing status on stdout, or display a message on stdout

Format: ECHO [ON | OFF | message]

message Text to display.

See also the commands [ECHOS](#), [ECHOSERR](#), [ECHOERR](#), [ECHOX](#), [ECHOXERR](#), [SCREEN](#), [SCRPUT](#), [TEXT](#) and [VSCRPUT](#), and the internal variable [ECHO](#).

Usage:

The ECHO command has two unrelated, independently functioning purposes:

- [Command line echoing](#)
- [Message display](#)

Command line echoing

When command line echoing is enabled, each command is displayed on stdout after it is fully parsed, aliases, functions, and variables expanded, but before it is executed.

Echoing control

TCC controls command line echoing in batch files and at the interactive prompt independently.

Executing ECHO ON at the command prompt enables, and ECHO OFF disables echoing at the command prompt. ECHO defaults to OFF at the command line. The command-line ECHO is most useful when you are learning how to use advanced features.

Similarly, executing ECHO ON in a batch file enables, and ECHO OFF disables echoing of batch file commands. ECHO defaults to ON in batch files. The current ECHO state is inherited by called batch files. You can change the default setting to OFF with the [SETDOS /V0](#) command, or the [Batch Echo](#) configuration option.

Regardless of the relevant echoing state, any command prefixed with the at-sign @ will not be echoed.

Echoing state display

To see the current echoing state, use the ECHO command with no parameters. This displays either the batch file or command line echo state, depending on where the ECHO command is performed.

Alternately, you can examine the value of the internal variable [ECHO](#).

Message display

If the ECHO command has a message (the whole command tail, excluding redirection or piping, if any), and message is neither of the words ON or OFF (though it can include those words), message is fully parsed, then displayed on stdout, regardless of the applicable echoing state. Any display sent to stdout after message has been displayed will start on a new line.

Display rules

- The first space after the command name is ignored.
- Trailing spaces in **message** are ignored.
- Functions and variables not enclosed between back quotes are evaluated.
- To include special characters, e.g., < | >, in **message**, enclose them in double quotes or back quotes (see [Parameter Quoting](#)) or precede them with the [escape character](#), or use the /X option of the [SETDOS](#) command.
- To display % you may alternately use two % marks for each one to be displayed, e.g., %%
- To display trailing spaces, either enclose them in back quotes, or append a pair of back quotes behind them, e.g., **echo trailers ``**
- The [ASCII NUL](#) character cannot be included in **message**.
- If stdout is the console, after displaying **message** on the current line, the cursor will be moved to the beginning of the next line.
- If stdout is a file, the **CR LF** sequence will be appended to **message**.

To display a blank line, use one of the forms below:

```
echo ``      (two consecutive back quotes), or
echo .      (special syntax for compatibility with CMD).
```

Examples

This command will display a message:

```
echo Processing your print files...
```

The command

```
echo      This text is indented 3 spaces ``
```

will display 3 leading and 3 trailing spaces.

4.3.43 ECHOERR

Purpose: Display a message to the standard error device (stderr)

Format: ECHOERR message

message Text to display.

See also: [ECHO](#), [ECHOS](#), [ECHOSERR](#), [ECHOX](#), and [ECHOXERR](#).

Usage:

ECHOERR (like [ECHO](#) in message display mode) parses and expands **message**, and displays it on stderr (usually the screen), instead of stdout. Even if stdout of a batch file is redirected or piped, ECHOERR will still display a screen message, unless stderr is redirected or piped (see [Redirection](#)). Any display sent to stderr after **message** has been displayed will start on a new line.

Display rules

- The first space after the command name is ignored.
- Trailing spaces in **message** are ignored.
- Functions and variables not enclosed between back quotes are expanded.
- To include special characters, .e.g, < | >, in **message**, enclose them in double quotes or back quotes (see [Parameter Quoting](#)) or precede them with the [escape character](#), or use the /X option of the [SETDOS](#) command.
- To display % you may alternately use two % marks for each one to be displayed, e.g., %%
- To display trailing spaces, either enclose them in back quotes, or append a pair of back quotes behind them, e.g., **echoerr trailers** ``
- The [ASCII NUL](#) character cannot be included in **message**.
- If stderr is the console, after displaying **message** on the current line, the cursor will be moved to the beginning of the next line.
- If stderr is a file, the CR LF sequence will be appended to **message**.

4.3.44 ECHOS

Purpose: Display a message to standard output (stdout) without a trailing carriage return / line feed

Format: ECHOS message

message Text to display.

See also: [ECHO](#), [ECHOERR](#), [ECHOSERR](#), [ECHOX](#), [ECHOXERR](#), [SCREEN](#), [SCRPUT](#), [TEXT](#), and [VSCRPUT](#).

Usage:

ECHOS, like [ECHO](#) in message display mode, parses, expands, and displays **message** on stdout. However, any display sent to stdout after **message** has been displayed will continue on the same line.

Display rules

- The first space after the command name is ignored.
- Trailing spaces in **message** are ignored.
- Functions and variables not enclosed between back quotes are evaluated.
- To include special characters, .e.g, < | >, in **message**, enclose them in double quotes or back quotes (see [Parameter Quoting](#)) or precede them with the [escape character](#), or use the /X option of the [SETDOS](#) command.
- To display % you may alternately use two % marks for each one to be displayed, e.g., %%
- To display trailing spaces, either enclose them in back quotes, or append a pair of back quotes behind them, e.g., **echo trailers** ``
- The [ASCII NUL](#) character cannot be included in **message**.
- ECHOS keeps the cursor on the same line, thus permitting building a line of display using multiple commands

ECHOS is useful for text output when you don't want to add a carriage return / linefeed pair at the end of the line. This is useful if your whole line of text requires more than one command to build, and also for controlling character devices.

4.3.45 ECHOERR

Purpose: Display a message to the standard error device (stderr) without a trailing carriage return / line feed

Format: ECHOERR message

message Text to display.

See also: [ECHO](#), [ECHOS](#), and [ECHOERR](#).

Usage:

ECHOERR acts as a combination of [ECHOS](#) and [ECHOERR](#). It parses and expands **message**, and displays it on stderr. However, any display sent to stderr after **message** has been displayed will continue on the same line.

Display rules

- The first space after the command name is ignored.
- Trailing spaces in **message** are ignored.
- Functions and variables not enclosed between back quotes are evaluated.
- To include special characters, e.g., < | >, in **message**, enclose them in double quotes or back quotes (see [Parameter Quoting](#)) or precede them with the [escape character](#), or use the /X option of the [SETDOS](#) command.
- To display % you may alternately use two % marks for each one to be displayed, e.g., %%
- To display trailing spaces, either enclose them in back quotes, or append a pair of back quotes behind them, e.g., echo trailers ``
- The [ASCII NUL](#) character cannot be included in **message**.
- ECHOERR keeps the cursor on the same line, thus permitting building a line of display using multiple commands

4.3.46 ECHOX

Purpose: Display a message to standard output (stdout) without performing any variable expansion or redirection.

Format: ECHOX message

message Text to display.

See also: [ECHO](#), [ECHOERR](#), [ECHOSERR](#), [ECHOXERR](#), [SCREEN](#), [SCRPUT](#), [TEXT](#), and [VSCRPUT](#).

Usage:

ECHOX will echo the message text to STDOUT without doing any of the parser processing (variables, redirection, escaped characters, etc.).

Display rules

- The first space after the command name is ignored.
- Trailing spaces in **message** are ignored.
- The ASCII NUL character cannot be included in **message**.

ECHOX is useful for text output when you want to display some text that may have embedded special characters (like %, <, >, or |).

4.3.47 ECHOXERR

Purpose: Display a message to standard error (STDERR) without performing any variable expansion or redirection.

Format: ECHOXERR message

message Text to display.

See also: [ECHO](#), [ECHOERR](#), [ECHOSERR](#), [ECHOX](#), [SCREEN](#), [SCRPUT](#), [TEXT](#), and [VSCRPUT](#).

Usage:

ECHOXERR will echo the message text to STDERR without doing any of the parser processing (variables, redirection, escaped characters, etc.).

Display rules

- The first space after the command name is ignored.
- Trailing spaces in **message** are ignored.
- The ASCII NUL character cannot be included in **message**.

ECHOXERR is useful for text output when you want to display some text that may have embedded special characters (like %, <, >, or |).

4.3.48 EJECTMEDIA

Purpose: Eject removable media in the specified drive(s)

Format: EJECTMEDIA drive ...

Usage:

EJECTMEDIA will eject removable media, such as CD-ROMs, DVDs, etc. (It is not intended for unmounting USB drives.)

See also [LOADMEDIA](#).

4.3.49 ENDLOCAL

Purpose: Restore the saved disk drive, directory, environment, local alias and function lists, and special characters, and exports selected variables

Format: ENDLOCAL [/D] [exportvar ...]

[/D\(ont restore\)](#)

See also: [SETLOCAL](#).

Usage:

The [SETLOCAL](#) command saves the current disk drive, default directory, all environment variables, the alias and function lists, and the command separator, escape character, parameter character, decimal separator, and thousands separator. It does not save the user-defined function list or array variables. ENDLOCAL restores everything that was saved by the previous [SETLOCAL](#) command, except as described below.

For example, this batch file fragment saves everything, removes all aliases so that user aliases will not affect batch file commands, changes the disk and directory, changes the command separator, runs a program, and then restores the original values:

```
setlocal
unalias *
cdd d:\test
setdos /c~
program ~ echo Done!
endlocal
```

[SETLOCAL](#) / ENDLOCAL may be nested within a single batch file up to 16 levels deep. You can also have multiple, separate [SETLOCAL](#) / ENDLOCAL pairs within a batch file, and nested batch files can each have their own [SETLOCAL](#) / ENDLOCAL. If you do not provide an ENDLOCAL in the batch file, **TCC** will do it automatically when the batch file exits.

You can also use [SETLOCAL](#) and ENDLOCAL in an alias or at the command line. The maximum nesting level from a command line or alias is 10 levels. Unlike batch files, you are responsible for matching the [SETLOCAL](#) / ENDLOCAL calls from an alias or command line; **TCC** will not perform an automatic ENDLOCAL.

An ENDLOCAL is performed automatically at the end of a batch file, or when returning from a "[GOSUB](#) filename". If you invoke one batch file from another without using [CALL](#), the first batch file is terminated, and an automatic ENDLOCAL is performed; the second batch file inherits the settings as they were prior to any [SETLOCAL](#).

• **Exporting environment variables**

The environment variables whose names are specified in the ENDLOCAL command are exported. This means that their names and values from inside the [SETLOCAL](#) / ENDLOCAL will be placed into the restored environment, either adding variables, or possibly modifying them. In the example below, the variable TEST will have the value **abcd** after the ENDLOCAL is executed, regardless of what its value was, or even if it had not been previously defined:

```
setlocal
set test=abcd
endlocal test
```

The list of variables to export may contain wildcards. All variables matching the requested pattern will be exported.

- **Exporting current working directory**

See option [/D](#) below.

Options:

- /D** (Don't restore directory) Export the current directory: the original drive and directory saved by [SETLOCAL](#) will not be restored.

4.3.50 ESET

Purpose: Edit an environment variable, alias or function definition

Format: ESET [/A /D /F /S /U /V] [/C var1 var2] name

name The name of an environment variable, function or alias to edit.

/A(alias)	/S(system variable)
/C(opy value)	/U(ser variable)
/Default environment)	/V(olatile variable)
/F(function)	

See also: [ALIAS](#), [FUNCTION](#), [SET](#), [UNALIAS](#), [UNFUNCTION](#), and [UNSET](#).

Usage:

ESET allows you to edit an environment variable, alias or function definition using line editing commands (see [Command Line Editing](#)).

For example, to edit the executable file search path:

```
eset path
path=c:\;c:\dos;c:\util
```

To create and then edit an alias:

```
alias d = dir /d/j/p
eset d
d=dir /d/j/p
```

Unless a specific data type is specified by one of the option switches **/A**, **/D**, **/F**, **/S**, **/U** or **/V**, ESET will search for **name** among environment variables first and then among aliases, thus if **name** is both a variable and an alias, ESET will edit the variable **name**, and ignore the alias **name**.

To edit variables defined in the Windows Registry or to edit functions, you **must** use the appropriate option switch.

Note: You cannot use ESET with [GOSUB variables](#).

If you have enabled global aliases (see [ALIAS](#)), any changes made to an alias with ESET will immediately affect all other copies of [TCC](#) which are using the same alias list. Similarly, if you have enabled global functions (see [FUNCTION](#)), any changes made to a function using ESET /F will

immediately affect all other copies of **TCC** which are using the same function list.

If you are editing an alias, ESET supports [filename completion](#).

Registry Variables: **Default**, **System**, **User**, and **Volatile** registry variables can be manipulated with the ESET command's **/D**, **/S**, **/U** and **/V** switches, respectively. For example, to edit volatile variable **myvar** from the registry, use:

```
eset /v myvar
```

Use caution when directly modifying registry variables as they may be essential to various Windows processes and applications.

Options:

/A Edit the named alias even if an environment variable of the same name exists. If you have an alias and an environment variable with the same name, you must use this switch to be able to edit the alias.

/C Copy the value from an existing variable, alias, or function. The syntax is:

```
eset /c var1 var2
```

where **var1** is the variable whose value you want to copy, and **var2** is the variable (new or existing) that you want to update.

/D Edit a "default" variable in the registry (HKU\DEFAULT\Environment).

/F Edit a user-defined function.

/S Edit a "system" variable in the registry (HKLM\System\CurrentControlSet\Control\Session Manager\Environment).

/U Edit a "user" variable in the registry (HKCU\Environment).

/V Edit a "volatile" variable in the registry (HKCU\Volatile Environment).

4.3.51 EVENTLOG

Purpose: Write a string to the Windows event log

Format: EVENTLOG [/S"source" /Cn /E /I /W] *message*

message The text to write.

source The source for this message.

[/C\(category\)](#)

[/S\(source\)](#)

[/E\(error\)](#)

[/W\(warning\)](#)

[/I\(informational\)](#)

See also: [HISTORY](#) and [LOG](#).

Usage:

EVENTLOG posts messages to the Windows application event log. You cannot use the [command separator](#) character ([&]) or the [redirection](#) symbols (| > <) in an EVENTLOG message, unless you enclose the message in [quotes](#) or precede the special characters with the [escape character](#).

By default, the text written with EVENTLOG is stored in the event log as informational messages. You can store warning and error messages by using the **/W** and **/E** switches.

Messages in the log can be reviewed with the Windows Event Log viewer.

If you do not have proper registry permissions when you execute the EVENTLOG command and/or the key cannot be created, EVENTLOG will fail and display an error. EVENTLOG is primarily intended for use by users with **Administrator** status. If you are running Windows Vista, 7, or 8, you will also need to be running an elevated session.

Options:

/= Display the EVENTLOG command dialog to help you set the command line options. You cannot specify any other arguments on the command line.

/Cn Set the event category. The value can be from 0-65535; Windows defines 0-7 as:

- 0 - None
- 1 - Devices
- 2 - Disk
- 3 - Printers
- 4 - Services
- 5 - Shell
- 6 - System
- 7 - Network

/E Store the message as an error entry in the event log.

/I Store the message as an informational entry in the event log. This is the default if no switch is used.

/S Specify the event log entry source. If you use the **/S** option, it must be the first option on the EVENTLOG command line. If the source contains white space, it must be double-quoted. For example:

```
eventlog /sCompiling /I Your message here.
```

/W Store the message as a warning entry in the event log.

4.3.52 EVENTMONITOR

Purpose: Monitor event logs

Format: EVENTMONITOR [/C [name]]
EVENTMONITOR server name /S"source" /T"type" /D"description" n command

server name	UNC name of the machine with the log file log name
------------------------	---

n Number of repetitions (or **FOREVER**)
command Command to execute when condition is triggered

<u>/C(ear)</u> <u>/D"description"</u>	<u>/S"source"</u> <u>/T"type"</u>
--	--------------------------------------

Usage:

If you don't enter any arguments, EVENTMONITOR will display the events it is currently monitoring.

The command line will be parsed and expanded before EVENTMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. EVENTMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

You can specify multiple **/D**, **/S**, and **/T** arguments. If you want to monitor multiple events in a log, put them into a single EVENTMONITOR command. EVENTMONITOR creates a separate thread for each EVENTMONITOR command, so if you have multiple commands you will be wasting CPU time, RAM, and risk having **command** executed simultaneously in different threads.

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

EVENTMONITOR creates environment variables when an event is triggered that can be queried by **command**. The variables are deleted after **command** is executed.

- _eventcomputer** The name of the computer than generated the event
- _eventcount** The number of times the condition has been triggered
- _eventdesc** The event description
- _eventlog** The name of the event log
- _eventsouce** The name of the source that wrote the event
- _eventtype** The event type (see [/T](#) below)

Options:

- /C** If **name** is specified, remove the monitor for that event. Otherwise, remove all event monitors.
- /D** Description for the event to be monitored. Only events with a matching description will set the trigger. The description may contain [regular expressions](#).
- /S** Source for the event to be monitored. Only events with a matching source will set the trigger. The source may contain [regular expressions](#).

/T Type of event to be monitored. Only events with a matching type will set the trigger. The types of events are:

Success
Error
Warning
Information
Audit_Success
Audit_Failure

4.3.53 EVERYTHING

Purpose: Search for files and/or directories

Format: EVERYTHING [/C /D /F /M=n /P /R /W @file] filename [...]

filename The file or directory name to search for
@file A text file containing the names of the files to search for, one per line (see [@file lists](#) for details)

[/C\(ase sensitive\)](#) [/P\(ath names\)](#)
[/D\(irectories only\)](#) [/R\(eular expression\)](#)
[/F\(iles only\)](#) [/W \(match whole word\)](#)
[/M=n \(maximum files\)](#)

File Selection

Supports [command dialog](#), extended [wildcards](#), and [multiple file names](#). Date, time, size or exclude ranges anywhere on the line apply to all *source* files. Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Usage

Search for files and/or directories on local NTFS drives using "Everything Search" (<http://www voidtools com>). EVERYTHING by default does a wildcard search equivalent to "**filename***", and outputs the full pathname of all matching files and/or directories.

You need to install Everything Search and index your local NTFS drives before using EVERYTHING.

Options

- /C** Filename matching is case sensitive
/D Only search for directories
/F Only search for files
/M=n Only return a maximum of *n* files / directories. (Note that /M determines the total number of matches prior to any additional filtering. If you use /D or /F you will end up with the total minus the number of directories or files you excluded.)
/P Match path names

/R *filename* is a regular expression (EVERYTHING will automatically set the regular expression flag if the filename begins with ::)

/W Match whole word

4.3.54 EXCEPT

Purpose: Perform a command on all available files except those specified

Format: EXCEPT [/I"text"] [(@file) | (file ...)] *command*

file The file or files to exclude from the command.

@file A text file containing the names of the files to exclude, one per line (see [@file lists](#) for details).

command The command to execute, including all appropriate parameters and switches.

[/I \(match description\)](#)

See also: [ATTRIB](#) and [File Exclusion Ranges](#).

File Selection

Supports extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#). Date, time, size, or file exclusion ranges must appear immediately after the EXCEPT keyword.

Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Usage:

EXCEPT provides a means of executing a command on a group of files and/or subdirectories, and excluding a subgroup from the operation. The **command** can be an internal command or alias, an external command, or a batch file.

You may use wildcards to specify the files to exclude from the **command**. The first example erases all the files in the current directory except those beginning with *MEMO*, and those whose extension is *.WKS*. The second example copies all the files and subdirectories on drive C to drive D except those in *C:\MSC* and *C:\TEST*, using the COPY command:

```
except (memo* *.wks) erase *
except (c:\msc c:\test) copy c:\* d:\ /s
```

When you use EXCEPT on an LFN drive, you must quote any file names inside the parentheses which contain white space or special characters. For example, to copy all files except those in the "Program Files" directory to drive E:\:

```
except ("Program Files") copy /s * e:\
```

EXCEPT will assume that the files to be excluded are in the current directory, unless another directory is specified explicitly.

EXCEPT prevents operations on the specified file(s) by setting the hidden attribute, performing the command, and then clearing the hidden attribute. If the command is aborted in an unusual way, you may

need to use the ATTRIB command to remove the hidden attribute from the file(s). Files which already had the hidden attribute, and are included in the set matching EXCEPT, will not be hidden after EXCEPT is completed. The hidden attribute of files not matching EXCEPT will not be changed.

Caution: EXCEPT will not work with programs or commands that ignore the hidden attribute or which work explicitly with hidden files, including [DEL](#) /Z, and the /H (process hidden files) switch available in internal file processing commands.

[File exclusion ranges](#) provide a faster and more flexible method of excluding files from internal commands, and do not manipulate file attributes, as EXCEPT does. However, exclusion ranges can only be used with internal commands; you must use EXCEPT for external commands.

Date, time, and size ranges can be used immediately after the word EXCEPT to further qualify which files should be excluded from the **command**. If the **command** is an internal command that supports ranges, an independent range can also be used in the **command** itself. You can also use a file exclusion range within the EXCEPT command; however, this will select files to be excluded from EXCEPT, and therefore included in execution of the **command**.

You can use [command grouping](#) to execute multiple **commands** with a single EXCEPT. For example, the following command copies all files in the current directory whose extensions begin with .DA, except the .DAT files, to the D:\SAVE directory, then changes the first two characters of the extension of the copied files to .SA:

```
except (*.dat) (copy *.da* d:\save & ren *.da* *.sa*)
```

If you use filename completion (see [Filename Completion](#)) to enter the filenames inside the parentheses, type a space after the open parenthesis before entering a partial filename or pressing Tab. Otherwise, the command line editor will treat the open parenthesis as the first character of the filename to be completed.

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. EXCEPT will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

Option:

/I"text" Select files by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the /I immediately, with no intervening spaces. You can select all filenames that have a description with /I"[?]", or all filenames that do not have a description with /I"\"". Do not use /I with @file lists. See [@file lists](#) for details.

4.3.55 EXIT

Purpose: Exit the current **TCC** session

Format: EXIT [/B] [value]

value The numeric exit code to return.

[/B \(exit from batch file\)](#)

Usage:

EXIT terminates the current copy of the command processor.

To close the session, or to return to the application that started the command processor, type:

```
exit
```

If you specify a value, EXIT will return that value to the program that started the command processor. For example:

```
exit 255
```

The *value* is a number you can use to inform the program of some result, such as the success or failure of a batch file. It can range from 0 - 4,294,967,295.

Option:

- /B Exit the current batch file, rather than the shell. This switch is for compatibility with CMD. The [CANCEL](#) and [QUIT](#) commands are generally more flexible for use in batch files.

4.3.56 FFIND

Purpose: Search for files by name or contents

Format: FFIND [/8 /A[:][-]rhsadecijopt /B /C /D[list] /E["text"] /F /G /H /I /"text" /K /L /Ln /M /N[dehjs] /O:[-]acdeginorsu /P /R /S[+[n] /[T]X"xx" /U /V /W /Y /+n /-n] file...

list A list of disk drive letters (without colons).

file The file, directory, or list of files or directories to display.

/[+ -] skip matches	/L(ine numbers or header/footer lines)
/8 (UTF-8)	/M (no footers)
/A(attribute select)	/N(ot)
/B(are)	/O(rder)
/C(ase sensitive)	/P(ause)
/D(rive)	/R(everse search order)
/E (upper case)	/S(ubdirectories)
/E"xx" (regular expression)	/T"xx" (text search string)
/F (stop after match)	/U (summary only)
/G (goto directory)	/V (verbose)
/H (ignore binary files)	/W (Find dialog)
/I(gnore wildcards)	/X"xx" (hex display / search string)
/I"text" (match description)	/Y (prompt to stop after match)
/K (no headers)	

File Selection

Supports extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

Internet: Can be used with [FTP Servers](#).

Usage:

FFIND is a flexible search command that looks for files based on their names and their contents. Depending on the options you choose, FFIND can display filenames, matching text, or a combination of

both in a variety of formats.

If you don't supply a file name, FFIND will read from standard input. (This allows you to pipe or redirect input to FFIND.)

If you want to search for files by name, FFIND works much like the DIR command. For example, to generate a list of all the *.BTM* files in the current directory, you could use the command

```
ffind *.btm
```

The output from this command is a list of full pathnames, followed by the number of files found.

For example, if you want to limit the output to a list of **.BTM* files which contain the string *color*, you could use this command instead:

```
ffind /t"color" *v.btm
```

The output from this command is a list of files that contain the string *color* along with the first line in each file that contains that string. By default, FFIND uses a case-insensitive search, so the command above will include files that contain *COLOR*, *Color*, *color*, or any other combination of upper-case and lower-case letters.

If you would rather see the last line of each file that contains the search string, use the */R* option, which forces FFIND to search from the end of each file to the beginning. This option will also speed up searches somewhat if you are looking for text that will normally be at the end of a file, such as a signature line:

```
ffind /r /t"Sincerely," *.txt
```

You can use **TCC extended wildcards** in the search string to increase the flexibility of FFIND's search. For example, the following command will find *.TXT* files which contain either the string *June* or *July*. It will also find *Juny* and *Jule*. The */C* option makes the search case-sensitive:

```
ffind /c/t"Ju[nl][ey]" *.txt
```

If you want to search for text that contains wildcard characters (*, ?, [, or]), you can use the */I* option to force FFIND to interpret these as normal characters instead of wildcards. The following command, for example, finds all *.TXT* files that contain a question mark:

```
ffind /i/t"??" *.txt
```

Sometimes you may need to search for data that cannot be represented by ASCII characters. You can use FFIND's */X* option to represent the search string in hexadecimal format (this option also changes the output to show hexadecimal offsets rather than text lines). With */X*, the search must be represented by pairs of hexadecimal digits separated by spaces (in the example below, 41 63 65 is the hex code for "Ace"):

```
ffind /x"41 63 65" *.txt
```

You can also search using regular expressions using the */E* option. See [Regular Expression Syntax](#) for supported expressions.

When you use FFIND on an LFN drive, you must quote any file names which contain white space or

special characters.

FFIND can also find files on FTP servers. For example:

```
ffind /t"Windows" ftp://ftp.microsoft.com/windows
```

You can also use the IFTP command to start an FTP session on a server, and then use an abbreviated syntax to specify the files and directories you want. For more information, see [Using FTP/HTTP Servers and IFTP](#).

Note that searching for text in files on FTP servers (as in the command above) will be slow as the data from each file searched must be retrieved from the server and transferred to your computer to be checked for the search string.

FFIND sets three internal variables:

%_ffind_matches	The number of matches
%_ffind_files	The number of files found
%_ffind_errors	The number of errors

Options:

- /8** The file is interpreted as UTF-8. (Not available in **TCC/LE**.)
- /A:** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow **/A:**.
You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /B** Display file names only and omit the text that matches the search. This option is only useful in combination with **/T** or **/X**, which normally force FFIND to display file names and matching text.
- /C** Perform a case-sensitive search. This option is only valid with **/T**, which defaults to a case-insensitive search. It is not needed with a **/X** hexadecimal search, which is always case-sensitive.
- /D** Search all files on one or more drives. If you use **/D** without a list of drives, FFIND will search the drives specified in the list of files. If no drive letters are listed, FFIND will search all of the current drive. You can include a list of drives or a range of drives to search as part of the **/D** option. For example, to search drives C:, D:, E:, and G:, you can use either of these commands:


```
ffind /dcdeg ...
ffind /dc-eg ...
```

Drive letters listed after **/D** will be ignored when processing file names which also include a drive letter. For example, this command displays all the **.BTM** files on C: and E:, but only the **.BAT** files on D:

```
ffind /s /dce *.btm d:\*.bat
```

- /E** Display filenames in upper case.
- /E"text"** Search for a [regular expression](#). The regular expression must be contained in double quotes if it contains spaces, punctuation, or wildcard characters. See also **/T**.
- /F** Stops the search after the first match.
- /G** Change to the directory where the match was found (must be used with **/F**).
- /H** Don't search binary files. By default, this includes .exe, .sys, .dll, .zip, and .chm extensions. You can define your own list by setting the "BINARY_FILES" environment variable. For example, to ignore .exe, .sys, and .dll files:
- ```
BINARY_FILES=.exe;.sys;.dll
```
- /I** Only meaningful when used in conjunction with the **/T "text"** option. Suppresses the recognition of wildcard characters in the search text. This option is useful if you need to search for characters that would normally be interpreted as wildcards: \*, ?, [, and ].
- /I"text"** Select filenames by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the **/I** immediately, with no intervening spaces. You can select all filenames that have a description with **/I"[?]\*"**, or all filenames that do not have a description with **/I"[]"**.
- /K** Suppress the display of the header or filename for each matching text line.
- /L** Include the line number for each text line displayed. FFIND numbers lines beginning with **1**. A new line is counted for every CR or LF character (FFIND determines automatically which character is used for line breaks in each file), or when line length reaches the [command line length limit](#), whichever comes first.
- /Ln** The number of leading and trailing lines to display on a match. Each successive group of lines in a file will be separated by a "---" header.
- /M** Suppress the footer (the number of files and number of matches) at the end of FFIND's display.
- /N** Reverse the meaning of the search, i.e., report only files which contain no match. Setting **/N** will also set **/B**, i.e. searches are on a file-by-file basis; FFIND cannot search for all lines without match.
- A **/N** with one or more of the following arguments has an alternate meaning:
- |          |                            |
|----------|----------------------------|
| <b>d</b> | Skip hidden directories    |
| <b>e</b> | Don't display errors.      |
| <b>h</b> | No headers                 |
| <b>j</b> | Skip junctions             |
| <b>s</b> | Don't display the summary. |
- /O** Set the sort order for the files that FFIND displays. (Not available in **TCC/LE**.)

You may use any combination of the following sorting options; if multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as

the secondary key, and so on:

- Reverse the sort order for the next option
- a** Sort names and extensions in standard ASCII order, rather than sorting numerically when digits are included in the name or extension
- c** Sort by compression ratio (the least compressed file in the list will be displayed first)
- d** Sort by date and time (oldest first); for drives which support long file names
- e** Sort by extension
- g** Group subdirectories first, then files
- i** Sort by file description (ignored if /O:c is also used)
- n** Sort by filename (this is the default)
- o** Sort by owner
- r** Reverse the sort order for all options
- s** Sort by size
- u** Unsorted

- /P** Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /R** Only meaningful when used in conjunction with the /T "text" or /X options. Searches each file from the end backwards to the beginning. This option is useful if you want to display the last occurrence of the search string in each file instead of the first (the default). It may also speed up searches for information that is normally at the end of a file, such as a signature.
- /S** Display matches from the current directory and all of its subdirectories. By default, FFIND processes only those subdirectories without the Hidden or System attributes. To view hidden or system subdirectories use /A along with /S.
 

If you specify a number following the /S, FFIND will limit the subdirectory recursion to the number specified. For example, if you have a directory tree "\a\b\c\d\e", /S2 will only go to the "a", "b", and "c" directories.

If you specify a + followed by a number after the /S, FFIND will not search for files until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\e, /S+2 will not find anything in \a or \a\b. (Not available in **TCC/LE**.)
- /T"text"** Specify the text search string. /T must be followed by a text string in double quotes (e.g., /t"color"). FFIND will perform a case-insensitive search unless you also use the /C option. For a hexadecimal search and/or hexadecimal display of the location where the search string is found, see /X. You can specify a search string with either /T or /X, but not both.
- /U** Display only the summary.
- /V** Show every matching line. FFIND's default behavior is to show only the first matching line, then to the next file. This option is only valid with /E, /T and /X.
- /W** Display the **Take Command Find Files** dialog. This option allows you to select the search options in a dialog instead of entering them on the command line. You cannot combine /W with any other FFIND options. (Not available in **TCC/LE**.)

**/X["xx.."]** Specify hexadecimal display and an optional hexadecimal search string.

If **/X** is followed by one or more pairs of hexadecimal digits in quotes (e.g., `/x"44 63 65"`), FFIND will search for that exact sequence of characters or data bytes without regard to the meaning of those bytes as text. If those bytes are found, the offset is displayed (in both decimal and hexadecimal). A search of this type will always be case-sensitive.

If **/X** is **not** followed by a hexadecimal search string it must be used in conjunction with **/E** or **/T**, and will change the output format to display offsets (in both decimal and hexadecimal) rather than actual text lines when the search string is found. For example, this command uses **/T** to display the first line in each BTM file containing the word "hello":

```
ffind /t"hello" *.btm
c:\test.btm:
echo hello

1 line in 1 file
```

If you use the same command with **/X**, the offset is displayed instead of the text:

```
ffind /t"hello" /x *.btm
c:\test.btm:
Offset: 1A

1 line in 1 file
```

You can specify a search string with either **/T** or **/X**, but not both.

- /Y** Prompt to stop searching after each match. This option is most useful when you are using FFIND to search for one specific file, and don't want to display all files which include a particular search string.
- /[+|-]n** **"/+n"** causes FFIND to skip the first **n** matches. **"/-n"** causes FFIND to stop after **n** matches.

#### 4.3.57 FIREWIREMONITOR

**Purpose:** Monitor FireWire device connection and disconnection

**Format:** FIREWIREMONITOR [/C [name]]  
FIREWIREMONITOR name CONNECTED | DISCONNECTED n command

|                |                                                |
|----------------|------------------------------------------------|
| <b>name</b>    | Device name                                    |
| <b>n</b>       | Number of repetitions (or <b>FOREVER</b> )     |
| <b>command</b> | Command to execute when condition is triggered |

[/C\(lear\)](#)

**Usage:**

The FireWire device name can include wildcards. You can use either the device ID or the "friendly" name

for the device.

The command line will be parsed and expanded before FIREWIREMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. FIREWIREMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing ).

If you don't enter any arguments, FIREWIREMONITOR will display the FireWire devices it is currently monitoring.

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

FIREWIREMONITOR creates three environment variables when a device is connected or disconnected that can be queried by **command**. The variables are deleted after **command** is executed.

- \_firewiredeviceid** The device ID (this may have special characters like & in the name, so you may need to use double quotes around the variable name to prevent **TCC** from parsing the special characters)
- \_firewirename** The "friendly" name of the device
- \_firewirecount** The number of times the condition has been triggered

#### **Options:**

- /C** If **name** is specified, remove the monitor for that FireWire device. Otherwise, remove all FireWire monitors.

### 4.3.58 FOLDERMONITOR

**Purpose:** Monitor folder and/or file creation, modification, and deletion

**Format:** FOLDERMONITOR [/C [folder]]  
 FOLDERMONITOR /S folder /I"file" /E"file" /U CREATED DELETED MODIFIED RENAMED  
*n* command

- folder** Folder (directory) or file name
- CREATED** Execute the command if the folder or file is created
- DELETED** Execute the command if the folder or file is deleted
- MODIFIED** Execute the command if the folder or file is modified
- RENAMEd** Execute the command if the folder or file is renamed
- n** Number of repetitions (or **FOREVER**)
- command** Command to execute when condition is triggered

- |                                    |                                          |
|------------------------------------|------------------------------------------|
| <a href="#"><u>/C(lear)</u></a>    | <a href="#"><u>/S(ubdirectories)</u></a> |
| <a href="#"><u>/E(xclude)</u></a>  | <a href="#"><u>/U(nlocked file)</u></a>  |
| <a href="#"><u>/I(include)</u></a> |                                          |

**Usage:**

If you don't enter any arguments, FOLDERMONITOR will display the folders and files it is currently monitoring, in the format:

| <b>folder</b> | <b>(include/exclude)</b> | <b>condition</b> | <b>(n)</b> | <b>command</b> |
|---------------|--------------------------|------------------|------------|----------------|
|---------------|--------------------------|------------------|------------|----------------|

The command line will be parsed and expanded before FOLDERMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

The **MODIFIED** condition is set if the file's size, attributes, or last access date and time are changed.

If you want to monitor multiple conditions for a file or folder, put them into a single FOLDERMONITOR command. FOLDERMONITOR creates a separate thread for each FOLDERMONITOR command, so if you have multiple commands you will be wasting CPU time, RAM, and risk having **command** executed simultaneously in different threads.

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. FOLDERMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing ).

When the condition is triggered, the command will be executed immediately in the separate thread. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

FOLDERMONITOR creates several environment variables when a file or folder is created, deleted, modified, or renamed that can be queried by **command**. The variables are deleted after **command** is executed.

**\_folderaction**    The type of change to the file or folder. The possible values are:

CREATED  
DELETED  
MODIFIED  
RENAME

**\_foldercount**    The number of times the condition has been triggered

**\_foldername**    The name of the folder being monitored

**\_folderfile1**    The name of the file or folder that was created/deleted/modified/rename. If the file was renamed, **folderfile1** is the old name.

**\_folderfile2**    If a file was renamed, **folderfile2** is the new name

For example, to monitor your **d:\results** directory and copy any new or modified files to a web page:

```
foldermonitor d:\results created modified forever copy "%__folderfile1%" "http://mycompany.com/results/"
```

**Options:**

- /C** If **name** is specified, remove the monitor for that folder. Otherwise, remove all folder / file monitors. **/C** cannot be combined with any other options.
- /E** Filename to be excluded. If you want to exclude multiple files, use multiple **/E** options. If you want to exclude a file in a specific subdirectory, the filename should include the relative path from the folder **name**. The name can include wildcards.
- /I** Filename to be included. If you want to include multiple files, use multiple **/I** options. If you want to include a file in a specific subdirectory, the filename should include the relative path from the folder **name**. The name can include wildcards.
- /S** Include subdirectories.
- /U** Don't set the trigger until the file is unlocked. Not compatible with CREATED, because FOLDERMONITOR will always get the notification before the file is accessible.

#### 4.3.59 FONT

**Purpose:** Change the console font

**Format:** FONT [/Ffamily /Nname /Wn /Xn /Yn]

|                       |                   |
|-----------------------|-------------------|
| <u>/F(ont family)</u> | <u>/X(width)</u>  |
| <u>/N(face name)</u>  | <u>/Y(height)</u> |
| <u>/W(eight)</u>      |                   |

**Usage:**

This command is only available in Windows Vista and later, and will only affect stand-alone TCC console windows. (You can change the font in Take Command tab windows using Configure Take Command / Tabs.)

**Options:**

- /F** The font family:  
decorative  
dontcare  
modern  
roman  
script  
swiss
- /N** The font face name.
- /W** The font weight (100 - 1000, on multiples of 100). The normal weight is 400; bold is 700.
- /X** The maximum width of a character, in logical units.
- /Y** The maximum height of a character, in logical units.

#### 4.3.60 FOR

**Purpose:** Repeat a command for several values of a variable

**Format:** File and string mode

```
FOR [range...] [/I"text"] [/A:[[-]+]rhsadecijopt /D /F ["options"] /H /Nj /O:[-]adegnrstu /R
[path] [/T"delimiters"] /W] %var IN (@set) DO command | (command ... [LEAVEFOR])
```

## Counted mode

```
FOR /L %var IN (start, step, end) DO command | (command ... [LEAVEFOR])
```

|                |                                                                               |
|----------------|-------------------------------------------------------------------------------|
| <b>options</b> | Parsing options for a "file parsing" FOR.                                     |
| <b>range</b>   | One or more <a href="#">range</a> specifications                              |
| <b>path</b>    | The starting directory for a "recursive" FOR.                                 |
| <b>%var</b>    | The variable to be used in the command ("FOR variable").                      |
| <b>set</b>     | A set of values for the variable.                                             |
| <b>start</b>   | The starting value for a "counted" FOR.                                       |
| <b>step</b>    | The increment value for a "counted" FOR.                                      |
| <b>end</b>     | The limit value for a "counted" FOR.                                          |
| <b>command</b> | A command or group of commands to be executed for each value of the variable. |

|                                        |                                     |
|----------------------------------------|-------------------------------------|
| <a href="#">/A: (Attribute select)</a> | <a href="#">/N (defaults)</a>       |
| <a href="#">/D(irectories only)</a>    | <a href="#">/O:... (Order)</a>      |
| <a href="#">/F(ile parsing)</a>        | <a href="#">/R(ecursive)</a>        |
| <a href="#">/H(ide dots)</a>           | <a href="#">/T (delimiter list)</a> |
| <a href="#">/I description range</a>   | <a href="#">/W(ildcards)</a>        |
| <a href="#">/L (counted loop)</a>      |                                     |

**File Selection**

Supports [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

Ranges must appear immediately after the FOR keyword after alias expansions (if any), and only affect the selection of files specified using wildcards.

Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

**Usage:**

FOR begins by creating a **set**. It then executes a command for every member of **set**. The command can be an internal command, an alias, an external command, or a batch file. The members of **set** can be a list of file names, text strings, a group of numeric values, or text read from a list of files.

When **set** is made up of text or several separate file names (not an include list), the elements must be separated by spaces, tabs, or commas.

FOR includes a large number of options, some of which duplicate functions available in other internal commands. It also supports additional conventions not found in our other commands, included for compatibility with CMD.

The first three sections below ([Working with Files](#), [Working with Text](#), and [Retrieving Text from Files](#)) describe the FOR command and the enhancements to it which are included in **TCC**. The sections on [Parsing Text from Files](#) and [Counted FOR Loops](#) describe features added for compatibility with CMD. The sections [Directory Recursion](#) and [Output Redirection](#) warn of special considerations. The section entitled [Other Notes](#) contains information you may need if you use any aspect of the FOR command extensively.

FOR sets two internal variables:

|              |                               |
|--------------|-------------------------------|
| %_for_files  | The number of files processed |
| %_for_errors | The number of errors          |

If the [Duplicate CMD Bugs](#) configuration option is set, **TCC** will emulate undocumented CMD behavior when FOR **set** arguments are split across multiple lines. For example:

```
for %a in (
one
two
three
) do (
echo %a
)
```

## Working with Files

Normally, **set** is a list of files specified with wildcards. For example, if you use this line in a batch file:

```
for %x in (*.txt) list %x
```

Then [LIST](#) will be executed once for each file in the current directory with the extension *.TXT*. The FOR variable **%x** is set equal to each of the file names in turn, then the LIST command is executed for each file. (You could do the same thing more easily with a simple LIST \*.TXT. We used FOR here so you could get a feel for how it operates, using a simple example. Many of the examples in this section are constructed in the same way.)

**Set** can include multiple files and include lists, like this:

```
for %x in (d:*.txt;*.doc;*.asc e:\test*.txt;*.doc) type %x
```

FOR supports [wildcards and extended wildcards](#), as well as [extended parent directory](#) names, e.g., ... \\*.txt to process all of the *.TXT* files that are contained in the directory 2 levels above the current directory.

By default those members of **set** that include wildcards match only files, not directories.

When you use FOR on an LFN drive, you must quote any file names within set which contain white space or special characters. The same restriction may apply to names returned in the FOR variable, if you pass them to **TCC** internal commands, or other commands which require quoting filenames with white space. FOR does not quote returned names automatically, even if you included quotes in set.

If set includes filenames, the file list can be further refined by using date, time, size, description and file exclusion [ranges](#). The range or ranges must be placed immediately after the word FOR. Ranges affect only those members of set which contain wildcards. For example, the FOR below will process all of the *.TXT* files that were created or updated on December 4, 2010, and of the file ABC.LST regardless of its timestamp:

```
for /[d12-4-2010,+0] %x in (*.txt abc.lst) ...
```

If **command** is an internal command that supports ranges, an independent range can also be used in

**command** itself.

You can also refine the list by limiting it with the [/A:](#) option to select only files that have specific attributes.

When you use wildcards to specify **set**, FOR scans the directory and finds each file which matches the wildcard name(s) you specified. If, during the processing of the FOR command, you create a new file that could be included in **set**, it may or may not appear in a some later iteration of the same FOR command. Whether or not the new file appears depends on its physical location in the directory structure. For example, if you use FOR to execute a command for all *.TXT* files, and the command also creates one or more new *.TXT* files, those new files may or may not be processed during the current FOR command, depending on where they are placed in the physical structure of the directory. This is a Windows constraint over which **TCC** has no control. Therefore, in order to achieve consistent results you should construct FOR commands which do not create files that could become part of set for the current command.

### Working with Text

**Set** can also be made up of text instead of file names. For example, to create three files named *file1*, *file2*, and *file3*, each containing a blank line:

```
for %suffix in (1 2 3) echo. > file%suffix
```

You can also use the names of environment variables as the text. This example displays the name and content of several variables from the environment (see the general discussion of the [Environment](#) for details on the use of square brackets when expanding environment variables):

```
for %var in (path prompt comspec) echo %var=%[%var]
```

### Retrieving Text from Files

If the name of a file in **set** is prefixed with @ ("at" sign), it is considered as an [@file list](#). FOR extracts each line from the file and places it in the FOR variable.

**Warning:** if the line contains characters which are syntactically significant for **TCC**, for example, one of the characters <"[]>, it may have undesirable effects. You may use the /X option of [SETDOS](#) to mitigate them.

If you use @CON as the filename, FOR will read from standard input (typically a redirected input file) or from a pipe. If you use @CLIP: as the filename, FOR will read any text available from the Windows clipboard. See [Redirection and Piping](#) for more information on these features.

See [@file list](#) for additional details.

### Parsing Text from Files

Another method of working with text from files is to have FOR parse each line of each file for you. To begin a file-parsing FOR, you must use the [/F](#) option and include one or more file names in set. When you use this form of FOR, the variable name must be a single letter, for example, %a.

This method of parsing, included for compatibility with CMD, can be cumbersome and inflexible. For a more powerful method, use FOR with [@filename](#) as the **set** to retrieve each line from the file, as described in the previous section, and use variable functions like [@FIELD](#), [@INSTR](#), [@LEFT](#), [@RIGHT](#),

and `@WORD` to parse the line (see [Variable Functions](#) for information on variable functions).

By default, FOR will extract the first word or token from each line and return it in the variable. For example, to display the first word on each line in the file *FLIST.TXT*:

```
for /f %a in (flist.txt) echo %a
```

You can control the way FOR /F parses each line by specifying one or more parsing options in a quoted string immediately after the `/F`. The available options are:

**skip=n**: FOR /F will skip *n* lines at the beginning of each file before parsing the remainder of the file.

**tokens=n, m, ...**: By default, FOR /F returns just the first word or **token** from each parsed line in the variable you named. You can have it return more than one token in the variable, or return tokens in several variables, with this option.

This option is followed by a list of numbers separated by commas. The first number tells FOR /F which token to return in the first variable, the second number tells it which to return in the second variable, etc. The variables follow each other alphabetically starting with the variable you name on the FOR command line. This example returns the first word of each line in TEST.TXT in `%d`, the second in `%e`, and the third in `%f`:

```
for /f "tokens=1,2,3" %d in (test.txt) ...
```

You can also indicate a range of tokens by separating the numbers with a hyphen -.

**eol=c**: If FOR /F finds the character *c* in the line, it will assume that the character and any text following it are part of a comment and ignore the rest of the line.

**delims=xxx..**: By default, FOR /F sees spaces, tabs and commas as word or token delimiters. This option replaces those delimiters with all of the characters following the equal sign to the end of the string. This option must therefore be the last one used in the quoted options string.

**usebackq** : Duplicates the awkward CMD syntax. A back quoted string is executed as a command; a single quoted string is a literal string; and double quotes quote filenames in the file set. We don't recommend **usebackq** for batch files written for **TCC**, as **TCC** has much more elegant ways of doing the same things.

You can also use FOR /F to parse a single string instead of each line of a file by using the string, in quotes, as **set**. For example, this command will assign variable **A** to the string **this**, **B** to **is**, etc., then display **this**:

```
for /f "tokens=1,2,3,4" %a in ("this is a test") echo %a
```

### "Counted" FOR Loop

The "counted FOR" loop is included for compatibility with CMD. In most cases, you will find the [DO](#) command more useful for performing counted loops.

In a counted FOR command, the **set** is made up of numeric values instead of text or file names. To begin a counted FOR command, you must use the `/L` option and then include three values, separated by commas, in **set**. These are the **start**, **step**, and **end** values. During the first iteration of the FOR loop, the variable is set equal to the **start** value. Before each iteration, the variable is increased by the **step** value.

The loop ends when the variable exceeds the **end** value. This example will print the numbers from 1 to 10:

```
for /l %val in (1,1,10) echo %val
```

This example will print the odd numbers from 1 to 10 (1, 3, 5, 7, and 9):

```
for /l %val in (1,2,10) echo %val
```

The **step** value can be negative. If it is, the loop will end when the variable is less than the **end** value.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

**WARNING!** You must not have white space between **start** and the subsequent comma, nor between **step** and its subsequent comma. White space after the comma is accepted.

### Directory Recursion

By default, FOR works only with files in the current directory or a specified directory. Option switch [/R](#) specifies that the search should recursively process subdirectories. If you specify a directory name immediately after [/R](#), FOR will start in that directory and then search each of its subdirectories. If no directory is specified after the [/R](#), the search starts in the current default directory. If you do specify a directory, and its name includes any special characters, it must be enclosed in double quotes. For example, it must be quoted if it is specified with the aid of an environment variable, e.g., `%windir%\command`.

There are two differences in the invocation of **command** caused by directory recursion:

- The loop control variable contains the full name of the matching file
- **command** is executed with the default directory set to the directory in which the file was found

This example processes all `.TXT` files in the current directory and its subdirectories:

```
for /r %x in (*.txt) ...
```

This example works with all of the `.BAK` files on drive **D**:

```
for /r d:\ %x in (*.bak) ...
```

### Output Redirection

The default output redirection (i.e., **for ... > filename**) creates a new output file in each iteration. If **filename** does not include an absolute file path, it will be created relative to the then current default directory. If you use directory recursion, this path will change for each directory processed. The simplest way to force a single target file is to enclose the whole command in parentheses, e.g.:

```
(for %x in (set) command) > filename
```

### Other Notes

- You can use either `%` or `%%` in front of the variable name (**var**) in the command. Either form will work, whether the FOR command is typed from the command line or is part of an alias or batch

file. (CMD which requires a single % if FOR is used at the command line, but requires %% if FOR is used in a batch file.) Note that you must have at least one % sign present.

- The variable name can be up to 80 characters long.
- If the FOR command is an alias, e.g., `alias for=*for /h, range` specifications will be ignored.
- The word DO is unnecessary but accepted. Do not confuse it with the completely unrelated [DO](#) command.
- If the name of the FOR variable **var** is a single character, for compatibility with CMD, it is created in the environment in a special way that does not overwrite an existing environment variable with the same name. Wherever **command** contains the % sign immediately followed by the character which is the name of the FOR variable, it is replaced by its value, regardless of any characters following it. For example, the following command tries to add **a:** and **b:** to the end of [PATH](#), but will not work as intended:

```
for %p in (a: b:) path %path;%p
path
b:ath;b:
```

The **%p** in **%path** was interpreted as the FOR variable **%p** followed by the text **ath**, not what was intended. To get around this, use a different letter or a longer name for the FOR variable, or use square brackets around the variable name, as shown in the examples below, any one of which accomplishes the original goal:

```
for %p in (a: b:) path %[path];%p
for %x in (a: b:) path %path;%x
for %px in (a: b:) path %path;%px
```

- If the name of the FOR variable contains more than one character, it is created in the environment, and erased when FOR is completed, whether or not a variable by that name existed before the FOR. It cannot be modified with the [SET](#), [ESET](#), or [UNSET](#) commands. If you already had a variable with that name, it will no longer be accessible. For example, a command that begins

```
for %path in ...
```

will write over your current [PATH](#) setting, then erase the [PATH](#) variable completely when FOR is done.

- **Command** may also use the FOR variable with the special syntax of CMD described in [Special syntax for CMD compatibility](#).
- The following example uses FOR with variable functions to delete the .BAK files for which a corresponding .TXT file exists in the current directory (this should be entered on one line):

```
for %file in (*.txt) del %@name[%file].bak
```

The above command may not work properly on an LFN drive, because the returned *FILE* variable might contain white space. To correct this problem, you need two sets of quotes, one for [DEL](#) and one for [%@NAME](#):

```
for %file in (*.txt) del "%@name[%file].bak"
```

- You can use [command grouping](#) to execute multiple commands for each element in [set](#). For example, the following command copies each .WKQ file in the current directory to the D:\WKS\SAVE directory, then changes the extension of each file in the current directory to .SAV:

```
[for %file in (*.wkq) (copy %file d:\wksave\ & ren %file *.sav)
```

or (in a batch file):

```
for %file in (*.wkq) (
 copy %file d:\wksave\
 ren %file *.sav
)
```

- In a batch file you can use [GOSUB](#) to execute a subroutine for every element in **set**. Within the subroutine, the FOR variable can be used just like environment variable. This is a convenient way to execute a complex sequence of commands for every element in **set** without [CALL](#)ing another batch file.
- One unusual use of FOR is to execute a collection of batch files or other commands with the same parameter. For example, you might want to have three batch files all operate on the same data file. The FOR command could look like this:

```
for %cmd in (filetest fileform fileprnt) %cmd datafile
```

This line will expand to three separate commands:

```
filetest datafile
fileform datafile
fileprnt datafile
```

- FOR statements can be nested.

## LEAVEFOR

The special keyword LEAVEFOR can be used inside a FOR command group. LEAVEFOR terminates the current FOR processing and continues with the line following the FOR command, in a manner similar to that of the LEAVE keyword in a [DO](#) command.

```
for %i in (*) (
 ...
 if "%i" == "xyz.abc" leavefor
 ...
)
```

### *Options:*

- /A:** Process only those files that have the specified attribute(s). **/A:** will be used only when processing wildcard file names in **set**. It will be ignored for filenames without wildcards or other items in **set**. See [Attribute Switches](#) for information on the attributes which can follow **/A:**.

For example, to process only those files with the archive attribute set:

```
for /a:a %f in (*) echo %f needs a backup!
```

Default: **/A:-D-H-S**, i.e. include only **files** without the **hidden** and **system** attributes.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

- /D** Only return subdirectories, excluding "." and "..".
- /F** Return one or more words or tokens from each line of each file in **set**. The **/F** option can be followed by one or more options in a quoted string which control how the parsing is performed. See [Parsing Text From Files](#).
- /H** Suppresses the assignment of the "." and ".." directories to the FOR variable when directories are explicitly included using the [/A:](#) option.
- /I"text"** Select filenames by matching text in their descriptions. See [Description Ranges](#).
- /L** Interpret the three values in **set** as the **start**, **step**, and **end** values of a counted loop. See [Counted FOR Loops](#).
- /Nj** Don't recurse into symlinks or junctions (see [/R](#)).
- /O:....** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see [/T:acw](#)
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

The **/O:....** option saves all of the matching filenames and then performs the requested operation. This avoids the potential problem of processing files more than once. (Not available in **TCC/LE**.)

**/R [path]** Look in the current directory and all of its subdirectories for files in **set**. If the **/R** is followed by a directory name, look for files in that directory and all of its subdirectories. **Warning:** if the directory name includes special characters, including "%" to indicate an environment variable, it must be enclosed in double quotes ("").

- /T"text"** Specify the delimiters to be used when parsing a string set.
- /W** The FOR set is to be processed as filenames, even if no wildcards are detected. (This is useful if you want to use regular expressions with FOR.)

### 4.3.61 FREE

**Purpose:** Display the total disk space, total bytes used, and total bytes free on the specified (or default) drive(s)

**Format:** FREE [*drive*: ... ]

*drive* One or more drives to include in the report.

See also: [MEMORY](#).

**Usage:**

A colon [:] is required after each drive letter. This example displays the status of drives A and C:

```
free a: c:
```

If the volume serial number is available, it will appear after the drive label or name.

FREE supports [OpenAFS](#) names.

### 4.3.62 FTYPE

**Purpose:** Modify or display the command used to open a file of a type specified in the Windows registry

**Format:** FTYPE [/P /R[filename] | filetype[=[command]]]

*filename* One or more input files to read file type definitions from.

*filetype* A file type stored in the Windows registry.

*command* The command to be executed when a file of the specified type is opened.

[/Pause](#)                    [/User](#)  
[/Read from file](#)

See also: [ASSOC](#), and [Executable Extensions](#).

**Usage**

FTYPE allows you to display or update the command used to open a file of a specified type stored in the Windows registry.

FTYPE modifies the behavior of Windows file associations stored under the registry handle HKEY\_CLASSES\_ROOT, and discussed in more detail under [Windows File Associations](#). If you are not familiar with file associations be sure to read about them before using FTYPE.

The entry modified by FTYPE is the Shell\Open\Command entry for the specified file type, which defines the application to execute when a file of that type is opened. The open action is generally invoked by selecting **Open** on the popup menu for a file from the Windows Explorer. Note that opening a file and double-clicking its icon (or selecting the icon and pressing Enter) may not be the same thing. Double-clicking or pressing Enter invokes the default action for the file type, which may or may not be **Open**.

If you invoke FTYPE with no parameters, it will display the current file types and associated shell open commands. Use the /P switch to pause the display at the end of each page. If you include a *filetype*,

with no equal sign or **command**, FTYPE will display the current command for that file type.

If you include the equal sign and **command**, FTYPE will create or update the shell open command for the specified file type. The **command** generally includes an application name, including full path, plus parameters. The specific syntax required depends on the internal operation of both Windows and the application involved, and is beyond the scope of this help file. You can learn about typical syntax by reviewing appropriate Windows and application documentation, and / or by checking through the current contents of your registry. If the value contains the percent mark character %, the value stored will be type REG\_EXPAND\_SZ, otherwise it will be type REG\_SZ.

To remove the shell open command for a file type, use a command like FTYPE **filetype=**, with no **command** parameter. This will not delete the shell open command entry from the registry; it simply sets the command to an empty string.

FTYPE should be used with caution, and only after backing up the registry. Improper changes to file associations can prevent applications and / or the operating system from working properly.

#### **Options**

- /P** Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /R** This option loads a list of file types and associated shell open commands. If no filename is specified and the input is redirected, FTYPE will read from stdin. The format of the file is the same as that of the FTYPE display.  
You can insert comments in the file by prefixing the line with a colon (:).
- /U** Display or set the association in HKCU\Software\Classes.

### **4.3.63 FUNCTION**

**Purpose:** Create, modify or display user-defined variable functions

**Format:** Display mode:  
FUNCTION [/G /L /P] [wildname]

Direct definition mode:  
FUNCTION [/G /L] name[=]definition

Definition file mode:  
FUNCTION [/G /L /O] /R [/Z] [file...]

|                   |                                                                                      |
|-------------------|--------------------------------------------------------------------------------------|
| <b>file</b>       | One or more input files to read function definitions from.                           |
| <b>wildname</b>   | Name of function whose definition is to be displayed (may contain * and ? wildcards) |
| <b>name</b>       | The name of the function you want to define.                                         |
| <b>definition</b> | The value or definition of what the function should return.                          |

[/G\(global\)](#)  
[/L\(ocal\)](#)  
[/P\(ause\)](#)

[/R\(ead file\)](#)  
[/Z \(overwrite list\)](#)

See also: [UNFUNCTION](#) and [ESET](#).

**Usage:**

- [Overview](#)
- [Displaying Functions](#)
- [Defining Functions](#)
- [Deleting Functions](#)
- [Local and Global Functions](#)
- [Saving and Reloading Your Functions](#)
- [Warnings](#)

## Overview

FUNCTION allows you to create or display user-defined variable functions that can be used anywhere [Variable Functions](#) can be used. User-defined functions are powerful alternatives to [subroutines](#).

## Displaying Functions

If you invoke the FUNCTION command with no parameters, it will display the current function list (the local function list if you have set local functions in *TCMD.INI* or the TCC startup command line; otherwise the global function list):

```
function
```

If you include a **wildname**, which may include wildcards (\*) or (?), with no equal sign and no **definition**, FUNCTION will display the current values, if any, of all functions matching **wildname**, .e.g.:

```
function *dx*
```

will display all functions which contain **dx** in their name.

You can use the [/P](#) option to control display scrolling when displaying functions.

## Defining Functions

If you include the equal sign and **definition**, FUNCTION will create or update the function referred to by **name**. Any previous **definition** associated with **name** is discarded. Instead of the = sign, you may use one or more spaces or tab characters to separate **name** and **definition**.

Once a function is defined, the definition may be edited using [ESET](#) /F.

A function can optionally use references to parameters numbered from %0 to %511 which will be replaced with the matching parameter value when the function is called. %0 refers to the function name, %1 to the first parameter, etc. For example, the function

```
function leftmost=%@left[1,%1]
```

will return the leftmost character in its parameter, e.g. %@**leftmost**[xyz] will return x.

The parameter %n\$ has a special meaning. **TCC** interprets it to mean "all arguments, from parameter **n** to the end." If **n** is not specified, it has a default value of 1, so %\$ means "all arguments passed to the

function."

The parameter `%-n$` means "the arguments from parameter 1 to *n* - 1".

The special variable reference `%#` expands to the number of parameters passed to the function.

A function definition need not reference any parameters at all. For example:

```
function tomorrow= `%@makedate[%@inc[%@date[%_date]]]`
```

could be simply invoked as `%@tomorrow[]`.

To use the function *name* you invoke is as `%@name[parameters]`, where you must specify enough parameters to assign a value to the highest numbered parameter **referenced** in the function definition. It may have more parameters, which will be silently ignored.

The [Colors, Color Names and Codes](#) topic shows a simple example of the use of a function in a batch file.

## Deleting Functions

The normal method is to use the [UNFUNCTION](#) command. However, it is also possible to delete a function by redefining it without a **definition**, e.g., the command

```
function fs=
```

deletes the function *fs*.

## Local and Global Functions

Functions can be stored in either a local or global list.

With a local function list, any changes made to the functions will only affect the current copy of **TCC**. They will not be visible in other shells or other sessions.

With a global function list, all copies of **TCC** will share the same function list, and any changes made to the functions in one copy will affect all other copies. This is the default in **TCC**.

You can control the type of function list with the [Local Functions](#) configuration option, with the `/L` and `/LF` options of the [START](#) command, and with the `/L` and `/LF` [startup options](#).

There is no fixed rule for determining whether to use a local or global function list. Depending on your work style, you may find it most convenient to use one type, or a mixture of types in different sessions or shells. We recommend that you start with the default approach, then modify it if you find a situation where the default is not convenient.

Whenever you start a second copy of **TCC** which uses a local function list, it inherits a copy of the functions from the previous shell. However, any changes to the functions made in the second shell will affect only that shell. If you want changes made in the second shell to affect the previous shell, use a global function list in both shells.

## Saving and Reloading Your Functions

You can save your functions to a file (e.g., *FUNCTIONS.LST*) this way:

```
function > function.lst
```

You can then reload all the function definitions in the file the next time you start up with the command:

```
function /r function.lst
```

This is much faster than defining each function individually in a batch file. If you keep your function definitions in a separate file which you load when **TCC** starts, you can edit them with a text editor, reload the edited file with FUNCTION /R, and know that the same function list will be loaded the next time you start **TCC**.

When you define functions in a file that will be processed by the FUNCTION /R command, you do not need back quotes around definition, even if back quotes would normally be required when defining the same function at the command line or in a batch file.

## Warnings

When you define a function in the command line (i.e., without using the */R* option), variables and functions not protected by back quotes or doubled % signs are immediately evaluated, and the result becomes part of the function definition.

Syntax errors in a function definition are not detected until it is used.

### Options:

- /G**      Switch from a local to a global function list.
- /L**      Switch from a global to a local function list.
- /O**      Don't overwrite existing values (only valid in combination with /R).
- /P**      Wait for a key to be pressed after each screen page before continuing the display.
- /R**      This option loads a list of functions from a file. If no filename is specified and input is redirected, /R will read from stdin. The format of the file is the same as that of the FUNCTION display:

*name=definition*

where **name** is the name of the function and **definition** specifies how to determine its value. You may use the equal sign = or whitespace to separate **name** and **definition**. Back-quotes are not required.

You can add comments to the file by starting each comment line with a colon :.

You can load multiple files with one FUNCTION /R command by placing the names on the command line, separated by spaces:

```
function /r func1.lst func2.lst
```

Each definition in a FUNCTION /R file can be up to 32,767 characters long. The definitions

can span multiple lines in the file if each line, except the last, is terminated with an [Escape Character](#).

If there is no filename parameter and input is redirected, FUNCTION /R will read from stdin.

- /Z Overwrite the existing function list with the contents of the specified file (must be used with /R). FUNCTION /R /Z is 20x faster than FUNCTION /R, because it doesn't have to check for existing functions and append new functions to the end of the list.

#### 4.3.64 GLOBAL

**Purpose:** Execute a command in the current directory and its subdirectories

**Format:** GLOBAL [/H /I /J /N /P /Q /S[+]n] command

**command** The command to execute, including parameters and switches.

|                                       |                                       |
|---------------------------------------|---------------------------------------|
| <a href="#">/H(idden directories)</a> | <a href="#">/P(rompt)</a>             |
| <a href="#">/I(gnore exit codes)</a>  | <a href="#">/Q(uiet)</a>              |
| <a href="#">/J (only junctions)</a>   | <a href="#">/S(ubdirectory depth)</a> |
| <a href="#">/N(o junctions)</a>       |                                       |

**Usage:**

GLOBAL performs **command** first in the current directory. Then it makes every subdirectory under the current directory the current working directory in turn, and performs **command** in that directory.

**Command** can be an internal command, an alias, an external command, or a batch file. When **command** is executed, it may be necessary to utilize one of the variable functions which convert a relative path to an absolute one, e.g., [@truename!](#), [@full!](#), etc to make sure that files of the same name in different directories are correctly handled.

If you don't specify any arguments, GLOBAL will display its command dialog.

The example below copies the files in every directory on drive **A** to the directory **C:\TEMP**:

```
[a:\] global copy * c:\temp
```

If a specific filename is found in more than one directory on **A**; assuming [COPY](#) is the default internal command, the one found last will be left in **C:\TEMP**. Which one of multiple, identically named files is found last is unpredictable!

If you use the [/P](#) option, GLOBAL will prompt for each subdirectory before performing **command**. You can use this option if you want to perform **command** in most, but not all subdirectories of the current directory.

You can use [command grouping](#) to execute multiple **commands** in each subdirectory. For example, the following command copies each **.TXT** file in the current directory and all of its subdirectories to drive **D**. It then changes the extension of each of the copied files to **.SAV**:

```
global (copy *.txt d: & ren *.txt *.sav)
```

#### Output Redirection

The default output redirection (i.e., **global command > filename**) creates a new output file named **filename** as each directory visited. If **filename** does not include an absolute file path, these files will be created relative to the currently visited directory. If **filename** does include an absolute file path, that file will be overwritten as each directory is visited, and only the data from the last visited directory will survive.

The simplest way to force a single target file is to enclose the whole command line in parentheses, e.g.:

```
(global command) > filename
```

**Options:**

- /=** Display the GLOBAL command dialog to help you set the command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /H** Forces GLOBAL to look for hidden directories. If you don't use this switch, hidden directories and their subdirectories are ignored without error indication.
- /I** If this option is not specified, GLOBAL will terminate if **command** returns a non-zero exit code. Use **/I** if you want **command** to continue in additional subdirectories even if it returns an error in one subdirectory. GLOBAL will normally halt execution if **TCC** receives a **Ctrl-C** or **Ctrl-Break** even if you use **/I**.
- Without this option, if GLOBAL is unable to change to a directory (for example, if user does not have access rights), GLOBAL will stop with an error message. With this option set, GLOBAL will ignore that directory, and all of its subdirectories, and continue in the next accessible directory.
- /J** Forces GLOBAL to only recurse through Junctions, not subdirectories.
- /N** Forces GLOBAL to ignore Junctions and only recurse through subdirectories.
- /P** Forces GLOBAL to prompt with each directory name before it performs **command** in that directory. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /Q** Do not display the directory names as each directory is processed.
- /S** GLOBAL will limit the subdirectory recursion to the number specified. For example, if you have a directory tree "\a\b\c\d\e", /S2 will only go to the "a", "b", and "c" directories.

If you specify a + followed by a number after the /S, GLOBAL will not execute **command** until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\e, /S+2 will not execute **command** in \a or \a\b. (Not available in **TCC/LE**.)

#### 4.3.65 GOSUB

**Purpose:** Execute a subroutine in the current batch file

**Format:** GOSUB ["filename"] label [variables]

- |                  |                                                          |
|------------------|----------------------------------------------------------|
| <b>filename</b>  | The file containing the subroutine                       |
| <b>label</b>     | The batch file label at the beginning of the subroutine. |
| <b>variables</b> | Optional GOSUB variables.                                |

See also: [CALL](#), [GOTO](#), and [RETURN](#).

**Usage:**

GOSUB can only be used in batch files.

TCC allows subroutines in batch files. A subroutine must start with a **label** (a colon [:] followed by a label name) which appears on a line by itself, and cannot be included a [command group](#). Case differences are ignored when matching labels. The subroutine must end with a [RETURN](#) statement.

The subroutine is invoked with a GOSUB command from another part of the batch file. After the RETURN, processing will continue with the command following the GOSUB command. For example, the following batch file fragment calls a subroutine which displays the directory and returns:

```
echo Calling a subroutine
gosub subr1
echo Returned from the subroutine
quit
:subr1
dir /a/w
return
```

GOSUB begins its search for the **label** on the line of the batch file immediately after the GOSUB command. If the **label** is not found between the current position and the end of the file, GOSUB will restart the search at the beginning of the file. If the label still is not found, the batch file is terminated with the error message "Label not found".

You can define GOSUB variables by placing them after the label name on the GOSUB line. For example:

```
Gosub Sub1 abc 15 "Hello World"
```

The variable names are defined on the label line. For example:

```
:Sub1 [str n world]
```

defines three variables - %str (set to "abc"), %n (set to 15), and %world (set to "Hello World"). Note that the square brackets are required on the label line. GOSUB variables are only defined for the duration of the subroutine. They are not inherited by nested GOSUBs, and are destroyed by the [RETURN](#) call.

If you define GOSUB variables on the label but do not supply them on the GOSUB line, they will be set to an empty string.

GOSUB calls with variables are limited to a maximum of 22 levels deep. There is no limit on normal GOSUB calls.

GOSUB variables are placed in the environment in a special form for the duration of the subroutine, and will "mask" any environment variables of the same name that existed before the subroutine was called. GOSUB variables can be referenced like normal environment variables, but are not stored in the same way, cannot be modified with the [SET](#), [ESET](#), or [UNSET](#) commands, and cannot be used with the DEFINED test of [IF](#), [IFF](#), or [@IF](#).

You cannot use [SET](#) within a subroutine to change the value of a GOSUB variable. If you attempt to do so, the SET command will set the standard environment variable of the same name, not the GOSUB

variable, but this value will be "masked" by the GOSUB variable and will remain inaccessible until the subroutine ends.

You can call a subroutine in another file by specifying ***filename*** (the name must be enclosed in double quotes, and the file cannot have been compressed with [BATCOMP](#)). This allows you to create libraries of subroutines, without having to duplicate them in each batch file. For example:

```
gosub "c:\library\batlib.btm" Evaluate [%1 %2 %3]
```

GOSUB saves the IFF and DO states, so IFF and DO statements inside a subroutine won't interfere with statements in the part of the batch file from which the subroutine was called. If the subroutine has executed a SETLOCAL without a matching ENDLOCAL, an ENDLOCAL will be executed before returning to the calling batch file.

You cannot [RETURN](#) from a GOSUB while inside a [DO](#) loop.

If **TCC** reaches the end of the batch file while inside a subroutine, it will automatically return to the command after the GOSUB, just as if an explicit [RETURN](#) command had been included as the last line of the file.

Subroutines can be nested.

See also: [user-defined functions](#).

#### 4.3.66 GOTO

**Purpose:** Branch to a specified line inside the current batch file

**Format:** GOTO [/I] *label*

*label*      The batch file label to branch to.

[/I\(FF and DO continue\)](#)

See also: [GOSUB](#), [CALL](#).

**Usage:**

GOTO can only be used in batch files.

After a GOTO command in a batch file, the next line to be executed will be the one immediately following the *label*. The *label* must begin with a colon [:] and appear on a line by itself, and cannot be included a [command group](#). The colon is required on the line where the *label* is defined, but is not required in the GOTO command itself. Case differences are ignored when matching labels.

This batch file fragment checks for the existence of the file CONFIG.SYS. If the file exists, the batch file jumps to C\_EXISTS and copies all the files from the current directory to the root directory on A:. Otherwise, it prints an error message and exits.

```
if exist config.sys goto C_EXISTS
echo CONFIG.SYS doesn't exist - quitting.
quit
:C_EXISTS
copy * a:\
```

GOTO begins its search for the ***label*** on the line of the batch file immediately after the GOTO command. If the ***label*** is not found between that position and the end of the file, GOTO will restart the search at the beginning of the file. If the label is still not found, the batch file is terminated with the error message "Label not found."

To avoid errors in the processing of nested statements and loops, GOTO cancels all active **IFF** statements and **DO** / ENDDO loops unless you use **/I**. This means that a normal GOTO (without **/I**) may not branch to any label that is between an IFF and the corresponding ENDIFF or between a DO and the corresponding ENDDO.

For compatibility with CMD, the command

```
GOTO :EOF
```

will end processing of the current batch file if the label :EOF does not exist. However, this is less efficient than using the **QUIT** or **CANCEL** command to end a batch file.

**Option:**

- /I** Prevents GOTO from canceling IFF statements and DO loops. Use this option only if you are absolutely certain that your GOTO command is branching entirely within any current IFF statement **and** any active DO / ENDDO block. Using **/I** under any other conditions will cause an error later in your batch file.

You cannot branch into another IFF statement, another DO loop, or a different IFF or DO nesting level, whether you use the **/I** option or not. If you do, you will eventually receive an "unknown command" error (or execution of the **UNKNOWN\_CMD** alias or plugin) on a subsequent ENDDO, ELSE, ELSEIFF, or ENDIFF statement.

#### 4.3.67 GZIP

**Purpose:** Create or update .gz (GZIP) archives

**Format:** **GZIP** [/A:[-][+]rhsdaecjot] /A /En /Ln /M /O:[-]adegnrstu /Q /V] **gziparchive** [@**file**] **file**

|                    |                                        |
|--------------------|----------------------------------------|
| <b>gziparchive</b> | The gzip file to work with             |
| <b>file</b>        | The files to be added to the gzip file |

|                                           |                                     |
|-------------------------------------------|-------------------------------------|
| <a href="#">/A:... (attribute switch)</a> | <a href="#">M(ove)</a>              |
| <a href="#">/A(dd)</a>                    | <a href="#">/O:... (sort order)</a> |
| <a href="#">/E (method)</a>               | <a href="#">/Q(uiet)</a>            |
| <a href="#">/Ln (compression level)</a>   | <a href="#">/V(iew)</a>             |

**File Selection**

Supports [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

**Usage:**

GZIP is compatible with the archives created by the Linux / UNIX gzip utility, and supports RFC 1952. GZIP is normally used for compressing a single file. If you need to compress multiple files, you should use the [ZIP](#) or [TAR](#) commands.

You can specify a pathname for **gziparchive**. If you don't provide an extension, and the filename as entered doesn't exist, GZIP adds ".gz". If you don't specify an operation, GZIP will default to Add.

**Option:**

**/A:...** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with **@file** lists. See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

**/A** Add the specified file to the archive. (This is the default.)

**/En** Set the compression method (0=deflate, 1=lzw). The default is 0.

**/Ln** Set the compression level (1 - 6, where 6=maximum compression). The default is 4.

**/M** Delete the file from the disk after adding them to the gzip file.

**/O:...** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the files will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

**/Q** Don't display the file being compressed.

**/V** View the contents of the .gz file (date, time, and filename). If the file was compressed with lzw, it will not have a header, so it cannot be viewed.

#### 4.3.68 HEAD

**Purpose:** Display the beginning of the specified file(s)

**Format:** HEAD [/A:[[-][+]rhsadecijopt] /B /Cn /l"text" /N[+]n /O:[-]adegrstu /P /Q /V] [**@file**] **file**...

**file** The file or list of files that you want to display.

**@file** A text file containing the names of the files to display, one per line (see [@file lists](#) for details).

|                                                                                                                                                    |                                                                                  |
|----------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------|
| <u>/A:</u> (Attribute select)<br><u>/B(ell)</u><br><u>/C</u> (number of bytes)<br><u>/I"text"</u> (match description)<br><u>/N(umber of lines)</u> | <u>/O:...</u> (Order)<br><u>/P(ause)</u><br><u>/Q(uiet)</u><br><u>/V(erbose)</u> |
|----------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------|

See also: [LIST](#), [TAIL](#), and [TYPE](#).

### **File Selection**

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

**Internet:** Can be used with [FTP/HTTP Servers](#), e.g.

```
head "http://jpsoft.com/notfound.htm"
```

### **Usage:**

The HEAD command displays the first part of a file or files. It is normally only useful for displaying ASCII text files (i.e. alphanumeric characters arranged in lines separated by CR/LF). Executable files (.EXE) and many data files may be unreadable when displayed with HEAD because they include non-alphanumeric characters or unusual line separators.

You can press **Ctrl-S** to pause HEAD's display and then any key to continue.

The following example displays the first 15 lines of the files *MEMO1* and *MEMO2*:

```
head /n15 memo1 memo2
```

To display text from the clipboard use **CLIP:** as the file name. CLIP: will not return any data if the clipboard does not contain text. See [Highlighting and Copying Text](#) for additional information on CLIP:.

HEAD sets two internal variables:

|                            |                               |
|----------------------------|-------------------------------|
| <code>%_head_files</code>  | The number of files displayed |
| <code>%_head_errors</code> | The number of errors          |

### **FTP Usage**

HEAD can also display files on [FTP/HTTP Servers](#). For example:

```
head ftp://ftp.microsoft.com/index
```

### **NTFS File Streams**

HEAD supports file streams on NTFS drives. You can type an individual stream by specifying the stream name, for example:

```
head streamfile:s1
```

### **Options:**

**/=** Display the HEAD command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.

**/A:** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow **/A:**. Do not use **/A:** with @file lists. See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

**/B** Ignore bell (ASCII 7) characters.

**/C:** Display the specified number of bytes. **/C** accepts a **b**, **k**, or **m** modifiers at the end of the number. **b** is the number of 512-byte blocks, **k** is the number of kilobytes, and **m** the number of megabytes.

**/I"text"** Select files by matching text in their descriptions. The text can include wildcards and extended wildcards. The search text must be enclosed in double quotes, and must follow the **/I** immediately, with no intervening spaces. You can select all filenames that have a description with **/I"[?]\*"**, or all filenames that do not have a description with **/I"\"**. Do not use **/I** with @file lists. See [@file lists](#) for details.

**/N+n** Skip the first **n** lines.

**/N n** Display **n** lines. The default is 10.

**/O....** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

**/P** Pause and prompt after displaying each page.

**/Q** Do not display a header for each file. This is the default behavior, but an explicit **/Q** may be needed to override an alias that forces **/V**.

**/V** Display a header for each file.

### 4.3.69 HELP

**Purpose:** Display help for internal commands

**Format:** HELP [topic]

**topic** A help topic (internal command, variable or function).

See also: [The Online Help System](#).

**Usage:**

Online help is available for all of **TCC**'s internal commands, variables, and other features.

The **TCC** help system (**tcmd.chm**) uses Microsoft's HTML Help Viewer (**HH.EXE**) included in all versions of Windows.

If you type the command **HELP** by itself (or press **F1** when the command line is empty), an introductory page (**Overview**) is displayed. If you type **HELP** plus a topic name, that topic is displayed. For example:

```
help copy
```

displays information about the **COPY** command and its options. All internal commands, internal variables, variable functions, and key mapping directives have their own topic.

You can also invoke help for the word immediately above (or immediately to the left of) the cursor by pressing [Ctrl-F1](#) (this can be useful when you need the syntax for a variable function).

### 4.3.70 HISTORY

**Purpose:** Display or modify the history list

**Format:** HISTORY [/A *command* /F["..."] /G /L /N /P /Rn *filename* /Tn /V]

**command** A command to be added to the history list.

**filename** The name of a file containing entries to be added to the history list.

|                                  |                                               |
|----------------------------------|-----------------------------------------------|
| <a href="#">/A(dd)</a>           | <a href="#">/P(ause)</a>                      |
| <a href="#">/F["..."] f(ree)</a> | <a href="#">/R(ead)</a>                       |
| <a href="#">/G(lobal)</a>        | <a href="#">/T (display last n lines)</a>     |
| <a href="#">/L(ocal)</a>         | <a href="#">/V (display in reverse order)</a> |
| <a href="#">/N(o duplicates)</a> |                                               |

See also: [DIRHISTORY](#), [HistoryExclude](#) and [LOG](#).

**Usage:**

**TCC** keeps a list of the commands you have entered on the command line. See [Command History and Recall](#) for information on command recall, which allows you to use the history list to repeat or edit commands you have previously executed.

The **HISTORY** command lets you view and manipulate the command history list directly. If no parameters are entered, **HISTORY** will display the current command history list.

With the options explained below, you can clear the list, add new commands to the list without executing them, save the list in a file, or read a new list from a file.

The number of commands saved in the history list depends on the length of each command line. The history list size can be specified at startup from 4,000 to 500,000 characters (see the [Command History Buffer Size](#) configuration option). The default size is 20,000 characters.

Your history list can be stored either locally (a separate history list for each copy of **TCC**) or globally (all copies of **TCC** share the same list). For full details see [local and global history](#).

You can use the HISTORY command as an aid in writing batch files by redirecting the HISTORY output to a file and then editing the file appropriately. However, it is easier to use the [LOG /H](#) command for this purpose.

You can disable the history list or specify a minimum command line length to save with the [Minimum Length](#) configuration option. You can prevent any command line from being saved in the history by beginning it with an "at" sign (@).

You can exclude specific commands from the History List with the [HistoryExclude](#) variable.

You can control whether duplicate entries will be saved in the history list with the [Duplicates](#) configuration option.

You can save the history list by redirecting the output of HISTORY to a file. This example saves the command history to a file called *HISTFILE* and reads it back again immediately. If you leave out the HISTORY /F command on the second line, the contents of the file will be appended to the current history list instead of replacing it:

```
history > histfile
history /f
history /r histfile
```

If you need to save your command history at the end of each day's work, you might use the first of these commands in your *TCSTART.BTM* or other startup file, and the second in *TCEXIT.BTM*:

```
if exist c:\histfile history /r c:\histfile
history > c:\histfile
```

This restores the previous history list if it exists, and saves the history when **TCC** exits.

**TCC** can also load and save the history list automatically if you use the [History File](#) configuration option.

#### **Options:**

**/=** Display the HISTORY command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

**/A** Add a command to the history list. This performs the same function as the **Ctrl-K** key at the command line.

**/F["..."]** Erase entries in the command history list. You can have multiple /F"..." arguments, and

they can contain wildcards. If you don't include the optional quoted argument, /F will erase the entire list.

- /G Switch from a local to a global history list.
- /L Switch from a global to a local history list.
- /N Removes duplicate entries (oldest first) from the history list.
- /P Wait for a key after displaying each page of the list. Your options at the prompt are explained in detail under [Prompts](#).
- /Rn Read the command history from the specified file and append it to the history list currently held in memory.

If you are creating a HISTORY /R file by hand, and need to create an entry that spans multiple lines in the file, you can do so by terminating each line, except the last, with an [escape character](#). However, you cannot use this method to exceed the command line length limit.

If you try to load a file that is larger than the history list size, HISTORY will only load the last part of the file that will fit.

You can optionally specify whether HISTORY should ignore duplicates and HistoryExclude and always append the lines by specifying /R1. (This will be considerably faster for large history lists.)

- /Tn Display the last *n* lines of the history. If *n* is negative, skip the first *-n* lines of the history. (Not available in [TCC/LE](#).)
- /V Display the history in reverse order. This cannot be combined with /T. (Not available in [TCC/LE](#).)

### 4.3.71 IF

**Purpose:** Execute a single command if a condition is true

**Format:**

- IF [/I] condition command
- IF [/I] condition (command1) ELSE (command2)

|                  |                                                      |
|------------------|------------------------------------------------------|
| <b>condition</b> | A <a href="#">conditional expression</a>             |
| <b>command</b>   | The command to execute if <b>condition</b> is TRUE.  |
| <b>command1</b>  | The command to execute if <b>condition</b> is TRUE.  |
| <b>command2</b>  | The command to execute if <b>condition</b> is FALSE. |

[/I\(gnore case\)](#)

See also: [Conditional expressions](#), [IFF](#), [@IF](#).

**Usage:**

IF is usually used only in aliases and batch files. It is always followed by a **condition** (see [Conditional expressions](#)), and then a **command**. First [condition](#) is evaluated, and if it is TRUE, **command** is

executed. Otherwise, **command** is ignored.

If the condition is FALSE, **IF** returns a non-zero result, so it can be evaluated by one of the conditional command operators (**||** or **&&**).

Do not use **IF** with multi-line **TCC** commands like **DO** (unless you use the single-line variant of **DO**).

The **IF ... ELSE ...** syntax of **CMD** is also supported:

```
IF [/I] condition (command1) ELSE (command2)
```

The commands to be executed must be enclosed in parentheses (as in a [command group](#)). If **condition** is TRUE, **command1** is executed, if FALSE, **command2** is executed. **Note:** this syntax is much less powerful than the [IFF](#) command, which is recommended.

If the last argument on the line is a single **,** it is interpreted as the beginning of a command group. **IF** will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing **.**

When an **IF** test fails, the remainder of the command is discarded. Whether **TCC** continues with the next command on the line, or discards the rest of the line and goes to the next line is dependent upon the [Duplicate CMD Bugs](#) configuration option. **CMD** will discard all remaining commands on the line when an **IF** test fails, including those after a command separator or pipe character. If you do not want to reproduce **CMD.EXE**'s behavior of an **IF** affecting all commands on a line, set **DuplicateBugs** to **No** in the **.INI** file. The **IF** behavior is different when **DuplicateBugs** is **YES** in a command group in a batch file. If there are multiple command lines in the command group, a failed **IF** will only ignore the remainder of the commands on that line. The commands on the subsequent lines in the command group will still be executed.

For example, if [Duplicate CMD Bugs](#) is enabled (the default), the following command will display nothing, because the second **ECHO** command is discarded along with the first when the condition fails. If [Duplicate CMD Bugs](#) is disabled, it will display "hello":

```
[c:\] if 1 == 2 echo Wrong! & echo hello
```

**Option:**

- /I This option is included only for compatibility with **CMD**. It has no effect in **TCC**, since all string comparisons are case-insensitive unless you specify a case-sensitive test (EQC).

### 4.3.72 IFF

**Purpose:** Perform one of several alternate sets of commands based on the values of conditional expressions

**Format:** **IFF** **condition1** **THEN**  
    **commandset1**  
    **[ELSEIFF** **condition2** **THEN**  
        **commandset2**]  
    ...  
    **[ELSE**  
        **commandset3**]  
**ENDIFF**

|                       |                                                                                               |
|-----------------------|-----------------------------------------------------------------------------------------------|
| <b>condition1,2,3</b> | <a href="#">Conditional expressions</a>                                                       |
| <b>commandset1</b>    | One or more commands to execute if <b>condition1</b> is TRUE                                  |
| <b>commandset2</b>    | One or more commands to execute if <b>condition1</b> is FALSE, but <b>condition2</b> is TRUE. |
| <b>commandset3</b>    | One or more commands to execute if both <b>condition1</b> and <b>condition2</b> are FALSE.    |

See also: [IF](#) and [@IF](#).

#### Usage:

IFF is similar to [IF](#), but it can perform one **commandset** when a [conditional expression](#) is true and a different **commandset** when it is false. Repeated use of the optional ELSEIFF clause permits IFF to sequentially evaluate multiple, independent [conditional expressions](#), and execute the **commandset** associated with the first TRUE [conditional expression](#), or, if none are true, the **commandset** associated with the optional ELSE clause. After execution of any one of the **commandsets** the command after the ENDIFF clause will be executed.

You must start a new line or include a [command separator](#):

- after each **THEN**
- before each **ELSEIFF**
- both before and after the **ELSE**.

The individual commands in each **commandset** may be *separate lines* of a batch file, or they may be separated by [command separators](#), in any combination. A **commandset** may also be empty. The individual commands in a **commandset** may include any internal command, alias, external command, or batch file.

IFF statements can be **nested**, i.e., a **commandset** may include another IFF / ENDIFF group. You must make sure that each individual command / **commandset** is syntactically correct. If an "inner" IFF / ENDIFF group is in error, it may not be detected until after the "outer" ENDIFF has been executed.

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. IFF will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing ).

#### Notes

Be sure to read the cautionary notes about [GOTO](#) and IFF under the [GOTO](#) command before using a [GOTO](#) inside an IFF statement.

If you [pipe](#) data to an IFF, the data will be passed to the command(s) following the IFF, not to IFF itself.

#### Example

The alias in this example checks to see if the parameter is a subdirectory. If so, the alias deletes the subdirectory's files and removes it (enter this on one line):

```
alias prune `iff isdir %1 then & del /s /x /z %1 & else & echo %1 is not
a directory! & endif`
```

#### 4.3.73 IFTP

**Purpose:** Open or close an FTP / FTPS / SFTP session

**Format:** IFTP [/= /S command /C /EP /IPv6 /N /Pn /PR="n" /Q /R /V /Z[n]] ["ftp://[user[:password]@]  
server[/path][:port]"]

**user** The user name to login to the FTP site  
**password** The password to login to the FTP site.  
**server** The FTP server name.  
**path** The default directory on the server for this session.  
**port** Port number.

|                               |                     |
|-------------------------------|---------------------|
| <u>/C(lose)</u>               | <u>/Q(uiet)</u>     |
| <u>/EP</u> (extended passive) | <u>/R(econnect)</u> |
| <u>/IPv6</u>                  | <u>/S(end)</u>      |
| <u>/N(o paths)</u>            | <u>/V(erbose)</u>   |
| <u>/P(assive)</u>             | <u>/Zn</u> (zlib)   |
| <u>/PR="n"</u> (port range)   |                     |

**Usage:**

Most file processing commands and functions in **TCC** can access files on FTP servers in the same manner as files on local hard drives and a local network. Normally, each time you use the FTP feature of one of these commands or functions, it repeatedly starts an FTP session, performs an individual operation, and closes the FTP session, until the command or function is finished.

IFTP starts an FTP session which remains open until you close it or it is closed by the remote server. There are several advantages to using IFTP: the FTP connection remains open so commands execute more quickly, the syntax for accessing files on the server is shorter, and you can specify a default directory on the server for file operations.

For example, to open an FTP connection using IFTP:

```
iftyp ftp://user:pwd@ftp.myserver.com/dir1
```

For an FTPS connection, use something like:

```
iftyp ftps://user:pwd@ftp.myserver.com/dir1
```

This command tells IFTP to open an FTP/FTPS session with the server **myserver.com**, send **user** as the login username and **password** as the login password, and to establish the directory **/dir1** as the default directory for this session. The user name and password are optional; if they are not used, IFTP will attempt to log in anonymously. Double quotes are required if there are spaces or special characters in the filename. If you specify a password of \*, you will be prompted to enter the password (which will appear on the screen as asterisks).

Note that in the example above **dir1** is a subdirectory of the FTP "root" directory -- the home directory for the named FTP user. In most server configurations this is not the same as the FTP server's physical root directory.

**Note:** If you enter IFTP with no parameters while a connection is active, the current server name and directory will be displayed.

If you enter IFTP with only the /Q or /V switch, you change the amount of information displayed without disturbing the existing connection (if any).

Once you have established an FTP session with IFTP, you can refer to files on the server by using **ftp:** (or **ftps:**) but leaving out the user name, password, and URL of the server. On most servers, file and path names which begin **ftp:** are relative to the default directory, if any, that you specified when you opened the IFTP session; file and path names which begin **ftp:/** are relative to the root directory for the login name.

The difference can be seen in these four [DIR](#) commands, assuming the IFTP session started above:

```
1. dir "ftp:*.txt"
2. dir "ftp:dir2/*.txt"
3. dir "ftp:/*.txt"
4. dir "ftp:/dir2/*.txt"
```

The first command lists the **.TXT** files in the default session directory, **dir1**. The second command lists the **.TXT** files in **/dir1/dir2** because it interprets the path **dir2/\*.txt** to be relative to the default directory. The quotes could be omitted from example 1 because it contains no forward slash that could be mistaken as an option switch. The third and fourth commands above, because they include a / immediately following the **ftp:** designator, are relative to the root directory. Command 3 lists the **.TXT** files in the root directory and command 4 lists the files in the **dir2** subdirectory of the root directory.

**Note:** If an ftp file or path specification begins with a ~ (tilde), **TCC** will not attempt to build a full directory name but will instead pass the entire string to the remote server.

You can only have one IFTP connection open at a time within a **TCC** tab window. However, while you have an IFTP connection open, you can still use a complete FTP URL to perform an operation on a different server. For example, while the session above is open, you can use this command to display all files in the root directory of **microsoft.com**:

```
dir "ftp://ftp.microsoft.com/*"
```

An IFTP session remains open until you explicitly close it with this command:

```
iftp /c
```

Most FTP servers "time out" after a period of inactivity. **TCC** will attempt to detect if the connection has been closed by the server, and reconnect if you reference the IFTP session again. You should not assume that an IFTP connection will continue to function if you leave it open but unused for a significant period of time. You can determine if the connection is still active with the [\\_iftp](#), [\\_ftps](#), and [\\_isftp](#) variables.

IFTP and the other FTP features of **TCC** rely on the server's compliance with Internet FTP standards. If your server is not fully compliant, or does not operate in the manner that **TCC** expects, commands may not work as you intend. We urge you to test each server you use with nondestructive commands like [DIR](#) before you try to copy or delete files, create or remove directories, etc.

Before you can use IFTP, you must establish the necessary connection to the Internet.

#### **Options:**

- /=** Display the IFTP command dialog to help you set the command line options. You cannot specify any other arguments on the command line.
- /C** Use this switch, with no URL, to close an IFTP session (see the example above).
- /EP** Use Extended Passive mode. (Works with FTP and FTPS, but not SFTP.)
- /IPv6** By default, IFTP expects an IPv4 address for the local and remote host, and will create an IPv4 socket. The /IPv6 option tells IFTP to use IPv6 instead. (Works with FTP, FTPS, and SFTP connections.)
- /N** Pass both source and target names to the server "as is" without any attempt at expanding the paths. This option should be used with caution and only for "non standard" servers for which the default processing fails to build a suitable name.
- /P** /P0 disables passive mode; /P1 enables it.
- /PR="n"** When using active mode, IFTP uses any available port to listen to incoming connections from the server. You can override this behavior by setting /PR (PortRange) to a value containing the range of ports the class will be listening to. The range is provided as *start-end*, for instance: "1024-" stands for anything higher than 1024, "1024-2048" stands for ports between 1024 and 2048 inclusive, "4000-4010, 50000-50010" stands for ports between 4000 and 4010 or between 50000 and 50010. (Works with FTP and FTPS, but not SFTP.)
- /Q** Turn off the display of the conversation with the FTP server.
- /R** Automatically reconnect if the FTP server times out.
- /S** Allows you to send commands directly to an FTP server. The connection must have already been opened by a previous IFTP command.
- /V** Display the dialog with the FTP server while opening the connection. This can be useful for debugging connection problems.
- /Zn** Use Zlib compression. You can optionally set the compression level (0-9; the default is 7). Zlib compression must be enabled on the server, and will only work with FTP and FTPS connections (not SFTP).

See [FTP Servers](#) for additional information on formatting and usage of FTP and FTPS references.

#### 4.3.74 INKEY

**Purpose:** Get a single keystroke from the user and store it in an environment or array variable

**Format:** INKEY [/C /D /E"n" /K"keys" /P /M /Wait /X] [prompt] %%varname

|                |                                                   |
|----------------|---------------------------------------------------|
| <b>prompt</b>  | Optional text that is displayed as a prompt.      |
| <b>varname</b> | The variable that will hold the user's keystroke. |
| <b>wait</b>    | Time to wait for a keystroke, in seconds          |

|           |                  |           |                    |
|-----------|------------------|-----------|--------------------|
| <b>/C</b> | Clear buffer     | <b>/P</b> | Password           |
| <b>/D</b> | Digits only      | <b>/W</b> | Wait               |
| <b>/K</b> | valid keystrokes | <b>/X</b> | no carriage return |

**/M** Mouse buttons

See also: [INPUT](#).

**Usage:**

INKEY optionally displays a prompt, then it waits for a specified time (or indefinitely) for a keystroke, and places the keystroke into an environment or [array](#) variable. It is normally used in batch files and aliases to get a menu choice or other single-key input. Along with the [INPUT](#) command, INKEY allows great flexibility in reading input from within a batch file or alias.

If **prompt** is included in an INKEY command, it is displayed while INKEY waits for input.

The following batch file fragment prompts for a character and stores it in the variable **NUM**:

```
inkey /D Enter a number from 1 to 9: %%num
```

INKEY reads standard input for the keystroke, so it will accept keystrokes from a redirected file or from [KEYSTACK](#). You can supply a list of valid keystrokes with the [/K](#) option.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

A standard keystroke is stored directly in the environment variable. An extended keystroke (for example, a function key or a and cursor key) is stored as a string, consisting of a leading @, followed by its scan code as a decimal number, e.g., the **F1** key is stored as **@59**. The **Enter** key is stored as an extended keystroke **@28**. See [ASCII, Key Codes, and ANSI X3.64 Commands](#) for scan codes.

When the [/M](#) option enables recognition of mouse buttons, (and [/W](#) is not specified), the variable is set to a single character with one of the codes below:

| <b>button</b> | <b>code</b> |
|---------------|-------------|
| left          | 240         |
| middle        | 498         |
| right         | 497         |

You can get the screen position of the last mouse click with the [\\_xmouse](#) and [\\_ymouse](#) internal variables.

To test for a non-printing value returned by INKEY use the [@ASCII](#) function to get the numeric value of the key, or convert the expected value of the code to a code using [@CHAR](#). For example, to test for **Esc**, which has an [ASCII](#) value of 27 or a left mouse button:

```
inkey Enter a key: %%key
if "%@ascii[%key]" == "27" echo Esc pressed
if %key EQ %@char[240] echo Left mouse button clicked
```

If you press **Ctrl-C** or **Ctrl-Break** while INKEY is waiting for a key, execution of an alias will be terminated, and execution of a batch file will be suspended while you are asked whether to cancel the batch job. A batch file can handle **Ctrl-C** and **Ctrl-Break** with the [ON BREAK](#) command.

INKEY works within the command line window. If you prefer to use a dialog for user input, see the [MSGBOX](#) and [QUERYBOX](#) commands.

**Options:**

- /C** Clears the keyboard buffer before INKEY accepts keystrokes. If you use this option, INKEY will ignore any keystrokes which you type, either accidentally or intentionally, before it is ready to accept input.
- /D** Only accept numbers from **0** to **9**.
- /K"keys"** Specifies the permissible keystrokes. The list of valid keystrokes should be enclosed in double quotes. For alphabetic keys the validity test is not case sensitive. You can specify extended keys by enclosing their names in square brackets (within the quotes), for example:

```
inkey /k"ab[Ctrl-F9]" Enter A, B, Ctrl-F9 %%var
```

See [Keys and Key Names](#) for a complete listing of the key names you can use within the square brackets, and a description of the key name format.

If an invalid keystroke is entered, **TCC** will echo the keystroke if possible, beep, move the cursor back one character, and wait for another keystroke.

- /M** Accept mouse button clicks. This is enabled only if Windows' Quick Edit is disabled (alt-space -> Properties -> Options).
- /P** Prevents INKEY from echoing the character.
- /W** Time-out period, in seconds, to wait for a response. If no keystroke is entered by the end of the time-out period, INKEY returns with the variable unchanged. This allows you to continue the batch file if the user does not respond in a given period of time. You can specify /W0 to return immediately if there are no keys waiting in the keyboard buffer. If /W is specified, mouse buttons are ignored.

For example, the following batch file fragment waits up to 10 seconds for a character, then tests to see if a "Y" was entered:

```
set netmon=N
inkey /K"YN" /w10 Network monitor (Y/N)? %%netmon
iff "%netmon" == "Y" then
 rem Commands to load the monitor program
endiff
```

- /X** Prevents INKEY from displaying a carriage return and line feed after the user's entry.

#### 4.3.75 INPUT

**Purpose:** Get a string from the keyboard and save it in an environment or array variable

**Format:** INPUT [/C /D /E["default"] /K"keys" /Ln /N /P /Wn /X] [*prompt*] %%*varname*

***prompt***      Optional text that is displayed as a prompt.  
***varname***     The variable that will hold the user's input.

|                                 |                                        |
|---------------------------------|----------------------------------------|
| <a href="#">/C(lear buffer)</a> | <a href="#">/N(o colors)</a>           |
| <a href="#">/D(igits only)</a>  | <a href="#">/Password)</a>             |
| <a href="#">/E(dit)</a>         | <a href="#">/W(ait)</a>                |
| <a href="#">/K(eys)</a>         | <a href="#">/X(no carriage return)</a> |
| <a href="#">/L(ength)</a>       |                                        |

See also: [SET](#), [INKEY](#), [KEYSTACK](#), [MSGBOX](#), and [QUERYBOX](#)

#### **Usage:**

INPUT optionally displays a prompt, then waits for your entry and stores it in an environment or [array](#) variable. INPUT is normally used in batch files and aliases to get multi-character input (for single keystroke input, see [INKEY](#)).

INPUT works within the command line window. If you prefer to use a dialog for user input, see the [MSGBOX](#) and [QUERYBOX](#) commands.

If **prompt** text is included in an INPUT command, it is displayed while INPUT waits for input. Standard command line editing keys may be used to edit the input string as it is entered. If you use the /P password option, INPUT will echo asterisks instead of the keys you type.

INPUT returns when you press carriage return. All characters entered up to, but not including, the carriage return are stored in the variable.

The following batch file fragment prompts for a string and stores it in the variable **FNAME**:

```
input Enter the file name: %%fname
```

INPUT reads standard input, so it will accept text from a redirected file or from the [KEYSTACK](#).

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

If you press Ctrl-C or Ctrl-Break while INPUT is waiting for input, execution of an alias will be terminated, and execution of a batch file will be suspended while you are asked whether to cancel the batch job. A batch file can handle Ctrl-C and Ctrl-Break itself with the [ON BREAK](#) command.

You can [pipe](#) text to INPUT, but it will set the variable in the "child" process used to handle the right hand side of the pipe. This variable will not be available in the original copy of **TCC** used to start the pipe.

#### **Options:**

- /C** Discard any keystrokes pending in the keyboard buffer before INPUT begins accepting characters.
- /D** Only accept numbers from 0 to 9.
- /E** Allows you to edit an existing value. If there is no existing value for **varname**, INPUT proceeds as if /E had not been used, and allows you to enter a new value. If there is no existing value and you provide an optional **default** value, INPUT will display the default value for editing (not available in **TCC/LE**).

**/K"keys"** Specifies the permissible keystrokes (not available in **TCC/LE**). The list of valid keystrokes should be enclosed in double quotes. For alphabetic keys the validity test is not case sensitive.

For example:

```
input /k"[0-9]-()" Enter your phone number: %%var
```

You can specify extended keys by enclosing their names in square brackets (within the quotes), See [Keys and Key Names](#) for a complete listing of the key names you can use within the square brackets, and a description of the key name format.

If an invalid keystroke is entered, **TCC** will beep and wait for another keystroke.

- /Ln** Sets the maximum number of characters which INPUT will accept to *n*. If you attempt to enter more than this number of characters, INPUT will beep and prevent further input (you will still be able to edit the characters typed before the limit was reached).
- /N** Disables the use of input colors defined in the [Colors](#) configuration options, and forces INPUT to use the default display colors.
- /P** Tells INPUT to echo asterisks, instead of the characters you type.
- /W** Time-out period, in seconds, to wait for a response. If no keystroke is entered by the end of the time-out period, INPUT returns with the variable unchanged. This allows you to continue the batch file if the user does not respond in a given period of time. If you enter a key before the time-out period, INPUT will wait indefinitely for the remainder of the line. You can specify **/W0** to return immediately if there are no keys waiting in the keyboard buffer.
- /X** Prevents INPUT from adding a carriage return and line feed after the user's entry.

#### 4.3.76 JABBER

**Purpose:** Send an IM via the JABBER network

**Format:** JABBER [/= /S"server" /U"user" / P"password" /Tn /V] /B target[@server] /F"filename" message

**message** The message to send

|                              |                             |
|------------------------------|-----------------------------|
| <a href="#">/B(uddy)</a>     | <a href="#">/Tn (port)</a>  |
| <a href="#">/F(ile)</a>      | <a href="#">/U(sername)</a> |
| <a href="#">/P(password)</a> | <a href="#">/V(erbose)</a>  |
| <a href="#">/S(erver)</a>    |                             |

**Usage:**

If /S, /U, and/or /P are not specified, JABBER will use the default values defined in the .INI file (JabberServer, JabberUser, and JabberPassword). If you don't specify any arguments, JABBER will display its command dialog.

JABBER is intended to send single short messages on an event (for example, when a large series of file transfers is completed), not as a general replacement for an interactive IM client.

Before using JABBER, you will need to create an account on a JABBER network server. See [www.jabber.org](http://www.jabber.org) for more information on the JABBER network and for open JABBER servers.

The JABBER command supports SSL, so it can talk with SSL XMPP servers (like talk.google.com).

**Options:**

- /= Display the JABBER command dialog to help you set the command line options. You cannot specify any other arguments on the command line.
- /B Address where the message will be sent
- /F Send a file to another user
- /P Logon password on the JABBER server
- /S JABBER server to log onto
- /T Server port (default 5222)
- /U User logon name on the JABBER server
- /V Display verbose (debugging) output

#### 4.3.77 KEYBD

**Purpose:** Set the state of the keyboard toggles Caps Lock, Num Lock, and Scroll Lock, or enable/disable the keyboard.

**Format:** KEYBD [/Cn /K[0|1] /Nn /Sn]

*n* can be either **0** to toggle the key off or **1** to toggle the key on.

|                                         |                                        |
|-----------------------------------------|----------------------------------------|
| <a href="#"><u>/C(aps lock)</u></a>     | <a href="#"><u>/N(um lock)</u></a>     |
| <a href="#"><u>/K(eyboard lock)</u></a> | <a href="#"><u>/S(scroll lock)</u></a> |

**Usage:**

Most keyboards have 3 toggle keys, the Caps Lock, Num Lock, and Scroll Lock. KEYBD lets you turn any toggle key on or off. It is most useful in batch files and aliases if you want the keys set a particular way before collecting input from the user.

For example, to turn off the Num Lock and Caps Lock keys, you can use this command:

```
keybd /c0 /n0
```

If you use the KEYBD command with no switches, it will display the present state of the toggle keys.

The toggle key state is typically the same for all sessions, and changes made with KEYBD in one session will therefore affect all other sessions.

**Options:**

- /C Turn the Caps Lock key on or off.

- /K** Disable (0) or enable (1) the keyboard. You can also reenable a disabled keyboard with Ctrl-Alt-End. (Not available in **TCC/LE**.)
- /N** Turn the Num Lock key on or off.
- /S** Turn the Scroll Lock key on or off.

#### 4.3.78 KEYS

**Purpose:** Enable, disable, or display the history list

**Format:** KEYS [ON | OFF | LIST]

See also: [HISTORY](#).

##### Usage

This command is provided for compatibility with KEYS command in CMD, which controls the history list in Windows. The same functions are available by setting the [Command History Minimum Length](#) configuration option, and by using the [HISTORY](#) command. (CMD's KEYS command no longer has an effect, because command line editing is always enabled.)

The history list collects the commands you type for later recall, editing, and viewing. You can view the contents of the list through the history list window or by typing any of the following commands:

```
history
history /p
keys list
```

The first command displays the entire history list. The second displays the entire list and pauses at the end of each full screen. The third command produces the same output as the first, except that each line is numbered.

You can disable the collection and storage of commands in the history list by typing:

```
keys off
```

You can turn the history back on with the command:

```
keys on
```

If you issue the KEYS command without any parameters, **TCC** will show you the current state of KEYS.

#### 4.3.79 KEYSTACK

**Purpose:** Send keystrokes to a program or command automatically

**Format:** KEYSTACK [/R filename] [/Wx] ["abc"] [keyname[n]] ...

- |                |                                                                        |
|----------------|------------------------------------------------------------------------|
| <b>/Wx</b>     | Delay in clock ticks before next insertion into the Keystack.          |
| <b>"abc"</b>   | Literal characters to be placed in the Keystack.                       |
| <b>keyname</b> | Name of a key whose code is to be placed in the Keystack or its ASCII. |
| <b>n</b>       | Number of times to repeat the immediately preceding <b>named</b> key.  |

[/R\(read file\)](#)[/W\(ait\)](#)**Usage:****Operation**

KEYSTACK takes a series of keystrokes and feeds them to a program or command as if they were typed at the keyboard. When the program has used all of the keystrokes in the keystack buffer, it will begin to read the keyboard for input, as it normally would.

KEYSTACK will send the keystrokes to the currently active window. If you want to send keystrokes to another program (rather than have them function with **TCC** itself), you must start the program or [ACTIVATE](#) its window so it can receive the keystrokes. You must do this before executing the KEYSTACK command.

KEYSTACK is most often used for programs started from batch files. In order for KEYSTACK to work in a batch file, you must start the program with the [START](#) command, then use the KEYSTACK command. If you start the program directly (without using [START](#)) the batch file will wait for the application to complete before continuing and running the KEYSTACK command, and the keystrokes will not appear in the target program.

If you use KEYSTACK in an alias executed from the prompt, the considerations are essentially the same, but depend on whether or not the [Wait for External Apps](#) configuration option is set. If it is **not** set (the default), you can use KEYSTACK immediately after an application is started. However, if [Wait for External Apps](#) is set, **TCC** will not execute any other operation until the program has finished, including the KEYSTACK command, and instead of the target program, the keystrokes will be sent to whatever program is running in the active window when KEYSTACK is executed.

You may not be able to use KEYSTACK effectively if you have programs running in the background which change the active window (for example, by popping up a dialog box). If a window pops up in the midst of your KEYSTACK sequence, keystrokes stored in the KEYSTACK buffer may go to that window, and not to the application you intended.

**Keystroke Interpretation**

Characters entered within double quotes (for example, "abc") will be sent to the target program as is. The only items allowed outside the quotes are key names, the [/W](#) option, and a repeat count. If you want to enter a double quote, use two double quotes. Do not prefix or append the two double quotes to a string argument.) For example, to insert the string abc "def"

```
keystack "abc" "" "def" ""
```

If **keyname** is a single letter, it is inserted in the keystack buffer as if it had been quoted, without any spaces. For example, you could enter the string abc as a b c, instead of the quoted string method described above.

If **keyname** is a number, it is interpreted as a virtual key code (0 - 255).

**Repetition.** To send **keyname** several times, follow it with a space, left bracket [, the repetition count, and a right bracket ]. For example, the command below will send the **Enter** key 4 times:

```
keystack enter [4]
```

The repeat count works only with an individual **keyname**. It cannot be used with quoted strings. You must have a blank space between the **keyname** and the repetition count.

See [Keys and key names](#) for a complete listing of key names and a description of the key name and numeric key code format.

#### Note

You may need to experiment with your programs and insert delays (see the [/W](#) option) to find the window activation and keystroke sequence that works for a particular program.

#### Example

To start Word and open the last document you worked on, you could use the command:

```
start word & keystack /w54 alt-f "1"
```

This starts **Word**, delays about three seconds (54 clock ticks at 1/18 second each) for **Word** to get started, places the keystrokes for **Alt-F** (**File** menu), and 1 (open the most recently used file) into the buffer. **Word** receives these keystrokes and performs the appropriate actions. Notice that the two commands, [START](#) and [KEYSTACK](#) are issued on a single command line. This ensures that the keystrokes are sent to **Word**'s window, not back to **TCC**.

#### Option:

- /R**      Read the KEYSTACK input from a file. (You can only read a single line.)
- /W**      Delay the next keystroke in the KEYSTACK buffer by a specified number of **clock ticks**. A clock tick is approximately 1/18 second. The number of clock ticks to delay should be placed immediately after the **W**, and must be between **1** and **65535** (65,535 ticks is about 1 hour). Do not use the Thousands Separator in the number! You can use the **/W** option as many times as desired and at any point in the string of keystrokes except within double quotes. Some programs may need the delays provided by **/W** in order to receive keystrokes properly from KEYSTACK. The only way to determine what delay is needed is to experiment.

### 4.3.80 LIST

**Purpose**      Display a text file, with forward and backward paging and scrolling

**Format**      LIST [range...] [/8 /A:[[-]+]rhsadecijopt /B[-]n /C /Etext" /F /H /I /L[-]n /N /O:[-]adegnrstu /R /S /T"text" /U /W /X[s]] [@file] [file...]

**file**      A file or list of files to display.

**@file**      A text file containing the names of the files to view, one per line (see [@file lists](#) for details).

**range**      A file selection [range](#) ([date](#), [description](#), [exclusion](#), [size](#), [time](#))

[/8 \(UTF-8\)](#)

[/A: \(Attribute select\)](#)

[/B\(YTE offset\)](#)

[/C \(separate console\)](#)

[/E \(regular expression \)](#)

[/N \(line numbers\)](#)

[/O:... \(Order\)](#)

[/R\(everse\)](#)

[/S\(standard input\)](#)

|                                            |                                              |
|--------------------------------------------|----------------------------------------------|
| <a href="#">/F (console screen buffer)</a> | <a href="#">/T (search for Text)</a>         |
| <a href="#">/H(igh bit off)</a>            | <a href="#">/U (Ruler)</a>                   |
| <a href="#">/I(gnore wildcards)</a>        | <a href="#">/W(rap)</a>                      |
| <a href="#">/L(ine offset)</a>             | <a href="#">/X(hexadecimal display mode)</a> |

See also: [VIEW](#), [HEAD](#), [TAIL](#), and [TYPE](#).

### **File Selection**

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

### **Internet**

Can be used with [FTP/HTTP Servers](#).

### **Usage**

(See the [VIEW](#) command for an updated and enhanced LIST replacement.)

LIST provides a fast and flexible way to view a file, without the overhead of loading and using a text editor.

For example, to display a file called *MEMO.DOC*:

```
list memo.doc
```

**Note:** LIST is primarily intended for displaying the contents of ASCII and Unicode text files (i.e. alphanumeric characters arranged in lines separated by CR/LF). It can be used for other files which contain non-alphabetic characters or unusual line separators, but you may need to use hexadecimal mode (see below) to display or search these files. Lines longer than 32,767 characters will be truncated unless you're in Wrap or Hex modes.

LIST displays files in the **TCC** window. If you resize the **TCC** window or the **Take Command** window when **TCC** is running in a tab window, LIST will automatically resize its display.

LIST recognizes the following keys and buttons:

| <b>Key (Button)</b>     | <b>Meaning</b>                                        |
|-------------------------|-------------------------------------------------------|
| <b>Home</b>             | Display the first page of the file                    |
| <b>End</b>              | Display the last page of the file                     |
| <b>Esc (ListExit)</b>   | Exit the current file                                 |
| <b>Ctrl-C (Quit)</b>    | Quit LIST                                             |
| <b>Ctrl-PgUp</b>        | Display previous file                                 |
| <b>Ctrl-PgDn</b>        | Display next file                                     |
| <b>Up Arrow</b>         | Scroll up one line                                    |
| <b>Down Arrow</b>       | Scroll down one line                                  |
| <b>Left Arrow</b>       | Scroll left 8 columns                                 |
| <b>Right Arrow</b>      | Scroll right 8 columns                                |
| <b>Ctrl Left Arrow</b>  | Scroll left 40 columns                                |
| <b>Ctrl Right Arrow</b> | Scroll right 40 columns                               |
| <b>Del</b>              | Prompt whether to delete the file                     |
| <b>Ins</b>              | Prompt whether to save the pipe or file to a new name |
| <b>Tab</b>              | Prompt for a new default tab size                     |

|                                                 |                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>F1</b>                                       | Display online help                                                                                                                                                                                                                                                                                                                                                                      |
| <b>F5</b> ( <a href="#">ListRefresh</a> )       | Refresh the display                                                                                                                                                                                                                                                                                                                                                                      |
| <b>B</b> ( <a href="#">ListPrevious</a> )       | Go back to the previous file in the current group of files                                                                                                                                                                                                                                                                                                                               |
| <b>Ctrl-B</b> ( <a href="#">ListClipboard</a> ) | Copy the current filename to the clipboard                                                                                                                                                                                                                                                                                                                                               |
| <b>C</b> ( <a href="#">ListContinue</a> )       | Continue with the next file                                                                                                                                                                                                                                                                                                                                                              |
| <b>e</b>                                        | Edit the file with the editor associated with that filetype. If there is no association, LIST will use the editor defined in the <a href="#">Editor</a> configuration option. If no editor is defined, LIST will use Notepad. If LIST is displaying a pipe, the contents are saved to the clipboard and the editor is started. (You will need to manually paste the clipboard contents.) |
| <b>F (Find)</b>                                 | Prompt and search for a string or a sequence of hexadecimal values                                                                                                                                                                                                                                                                                                                       |
| <b>Ctrl-F</b>                                   | Prompt and search for a string, searching backward from the end of the file                                                                                                                                                                                                                                                                                                              |
| <b>G (Goto)</b>                                 | Go to a specific line or, in hex mode, to a specific hexadecimal offset                                                                                                                                                                                                                                                                                                                  |
| <b>H (High)</b>                                 | Toggle the "strip high bit" ( <a href="#">H</a> ) option                                                                                                                                                                                                                                                                                                                                 |
| <b>I (Info)</b>                                 | Display information on the current file (the full name, size, date, and time)                                                                                                                                                                                                                                                                                                            |
| <b>L(in)e numbers</b>                           | Toggle the line numbering option                                                                                                                                                                                                                                                                                                                                                         |
| <b>N</b> ( <a href="#">ListNext</a> )           | Find next matching string                                                                                                                                                                                                                                                                                                                                                                |
| <b>Ctrl-N</b>                                   | Find previous matching string in the file                                                                                                                                                                                                                                                                                                                                                |
| <b>O</b> ( <a href="#">ListOpen</a> )           | Open a new file                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Ctrl-O</b>                                   | Open a new file                                                                                                                                                                                                                                                                                                                                                                          |
| <b>P (Print)</b>                                | Print selected pages or the entire file (make your selection in the Windows "Print" dialog)                                                                                                                                                                                                                                                                                              |
| <b>R</b> ( <a href="#">ListFindRegex</a> )      | Prompt and search for a <a href="#">regular expression</a>                                                                                                                                                                                                                                                                                                                               |
| <b>Ctrl-R</b>                                   | Prompt and search backwards for a <a href="#">regular expression</a>                                                                                                                                                                                                                                                                                                                     |
| <b>U</b> ( <a href="#">ListUnicode</a> )        | Toggle the Unicode display mode                                                                                                                                                                                                                                                                                                                                                          |
| <b>W (Wrap)</b>                                 | Toggle the "line wrap" ( <a href="#">W</a> ) option                                                                                                                                                                                                                                                                                                                                      |
| <b>X (Hex)</b>                                  | Toggle the hex-mode display ( <a href="#">X</a> ) option                                                                                                                                                                                                                                                                                                                                 |

Text searches performed with **F**, **N**, **Ctrl-F**, and **Ctrl-N**, or with the corresponding buttons, are not case-sensitive unless you check the Match case box in the search dialog. LIST remembers the search strings you have used in the current session; to select a previous string, use the drop-down arrow to the right of the string entry field (the **N** key and the Next button search for the top item in this drop-down list).

When the search string is found LIST displays the line containing the string at the top of the window, and highlights the string it found. Any additional occurrences of the string on the same display page are also highlighted. Highlighting is intended for use with text files. In binary files, the search string will be found but may not be highlighted properly.

If the display is currently in hexadecimal mode and you press **F** or **Ctrl-F**, you will be prompted for whether you want to search in hexadecimal mode. If so, you should then enter the search string as a sequence of 2-digit hexadecimal numbers separated by spaces, for example **41 63 65** ([ASCII](#) values for the string "Ace"). Hexadecimal searches are case-sensitive, and search for exactly the string you enter.

LIST saves the search string used by **F**, **N**, **Ctrl-F**, and **Ctrl-N** so you can LIST multiple files and search for the same string simply by pressing **N** in each file, or repeat your search the next time you use LIST.

You can use [extended wildcards](#) in the search string. For example, you can search for the string **to\*day** to find the next line which contains the word **to** followed by the word **day** later on the same line, or search for the numbers **101** or **401** with the search string **[14]01**. If you begin the search string with a back-quote **`**, or enclose it in back-quotes, wildcard characters in the string will be treated as normal text with no special wildcard meaning.

You can use the [/T](#) switch to specify search text for the first file. When you do so, LIST begins a search as soon as the file is loaded. Use [/I](#) to ignore wildcards in the initial search string, and [/R](#) to make the initial search go backwards from the end of the file. When you LIST multiple files with a single LIST command, these switches affect only the first file; they are ignored for the second and subsequent files.

You can also search using Regular Expressions using the **R** and **Ctrl-R** keys. See [Regular Expression Syntax](#) for supported expressions.

You can use the **G** key to go to a specific line number in the file (or to a specified hexadecimal offset in hex mode). LIST numbers lines beginning with 1. A new line is counted for every **CR** or **LF** character (LIST determines automatically which character is used for line breaks in each file), or when line length reaches 32,767 characters, whichever comes first.

LIST normally allows long lines in the file to extend past the right edge of the screen. You can use the horizontal scrolling keys (see above) to view text that extends beyond the screen width. If you use the **W** command or [/W](#) switch to wrap the display, each line is wrapped when it reaches the right edge of the screen, and the horizontal scrolling keys are disabled.

To view output from another command simply pipe the output of the command to LIST, for example:

```
dir | list
```

Normally LIST will detect input from a [pipe](#) automatically, but if it does not, use [/S](#) to explicitly specify piped input. Your ability to navigate backward through the displayed output (e.g. with **PgUp**) may be limited when viewing a large amount of data through a pipe, due to the way Windows handles piped output.

To view text from the clipboard, use **CLIP:** as the file to be listed. **CLIP:** will not return any data unless the clipboard contains text. See [Redirection](#) for more information on **CLIP:**

If you print the file which LIST is displaying, the print format will match the display format. If you have switched to hexadecimal or wrapped mode, that mode will be used for the printed output as well. If you print in wrapped mode, long lines will be wrapped at the width of the display. If you print in normal display mode without line wrap, long lines will be wrapped or truncated by the printer, not by LIST. Regardless of the display mode, LIST will bring up a standard Windows print dialog which allows you to print selected text, the current page, or the entire file.

- **FTP/HTTP Usage**

LIST can display files on [FTP servers](#) as well as the contents of HTTP/HTTPS URLs. For example:

```
list ftp://ftp.microsoft.com/index
list http://jpsoft.com/notfound.htm
```

You can also use the [IFTP](#) command to start an FTP session on a server, and then use an abbreviated syntax to specify the files and directories you want. For more information, see [Using FTP/HTTP Servers](#) and [IFTP](#).

- **NTFS File Streams**

LIST supports file streams on NTFS drives. You can list an individual stream by specifying the stream name, for example:

```
list streamfile:s1
```

If no stream name is specified the file's primary data is displayed.

See [NTFS File Streams](#) for additional details.

#### • Advanced Features

If you specify a directory name instead of a filename as a parameter, LIST will display each of the files in that directory.

If no filename is specified (and stdin is not redirected), LIST will open the common Windows "open file" dialog.

Most of the LIST keystrokes can be reassigned with [key mapping](#) directives.

By default, LIST sets tab stops every 8 columns. You can change this behavior with the [Tabs Width](#) configuration option.

#### Options

- /= Display the LIST command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /8 The file is interpreted as UTF-8. (Not available in **TCC/LE**.)
- /A: Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with [@file](#) lists. See [@file lists](#) for details.  
You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /B[-]n Start at byte **n**. If **n** is preceded by a minus sign -, start **n** bytes from the end of the file. The /B option will only display the file from the offset to the end; you cannot go back to a point before the offset.
- /C Display the file in a separate screen buffer and restore the original buffer upon exiting LIST. /C only works in stand-alone **TCC** windows, not in **Take Command** tab windows.
- /E Search for a [regular expression](#) in the first **file**. This option is the same as pressing **R**, but it allows you to specify the search text on the command line. The regular expression must be contained in double quotes if it contains spaces, punctuation, or wildcard characters. See also /T.
- /F Display the contents of the console screen buffer. (Not available in **TCC/LE**.)
- /H Strip the high bit from each character before displaying. This is useful when displaying files created by some word processors that turn on the high bit for formatting purposes. You can toggle this option on and off from within LIST with the **H** key or the tool bar.

- /I** Only meaningful when used in conjunction with the [/T](#) "text" option. Directs LIST to interpret characters such as \*, ?, [, and ] as literal characters instead of wildcard characters. /I affects only the initial search started by [/T](#), not subsequent searches started from within LIST.
- /I"text"** Select files by matching text in their descriptions. See [Description Ranges](#) for details.
- /L[-]n** Start at line **n**. If **n** is preceded by a minus sign -, start **-n** lines from the end of the file. The [/I](#) option only affects the initial page display; it does not prevent you from subsequently scrolling back to the start of the file.
- /N** Display line numbers. You can toggle the line numbers with the **L** key.
- /O:...** Sort the files before processing. (Not available in **TCC/LE**.)
- You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:
- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
  - Reverse the sort order for the next sort key
  - a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
  - d** Sort by date and time (oldest first); also see [/T:acw](#)
  - e** Sort by extension
  - g** Group subdirectories first, then files
  - r** Reverse the sort order for all options
  - s** Sort by size
  - t** Same as **d**
  - u** Unsorted
- /R** Only meaningful when used in conjunction with the [/T](#) "text" option. Directs LIST to search for text from the end of the file instead of from the beginning of the file. Using this switch can speed up searches for text that is normally near the end of the file, such as a signature. /R affects only the initial search started by [/T](#), not subsequent searches started from within LIST.
- /S** Read from standard input rather than a file. This allows you to redirect command output and view it with LIST. Normally, LIST will detect input from a redirected command and adjust automatically. However, you may find circumstances when /S is required. For example, to use LIST to display the output of [DIR](#) you could use either of these commands:
- ```
dir | list
dir | list /s
```
- /T** Search for text in the first **file**. This option is the same as pressing **F**, but it allows you to specify the search text on the command line. The text must be contained in double quotes if it contains spaces, punctuation, or wildcard characters. For example, to search for the string **TC** in the file **README.DOC**, you can use this command:

```
list /t"Take Command" readme.doc
```

The search text may include [wildcards and extended wildcards](#). For example, to search for the words **Hello** and **John** on the same line in the file **LETTER.DAT**:

```
list /t"Hello*John" letter.dat
```

When you display multiple files with a single LIST command, /T only initiates a search in the first file. It is ignored for the second and subsequent files. See also: [/I](#) and [/R](#).

- | | |
|-----------|--|
| /U | Display a ruler on the second line. |
| /W | Wrap the text at the right edge of the screen. This option is useful when displaying files that don't have a carriage return at the end of each line. The horizontal scrolling keys do not work when the display is wrapped. You can toggle this option on and off from within LIST with the W key or the Wrap button on the tool bar. |
| /X | Display the file in hexadecimal (hex) mode. This option is useful when displaying executable files and other files that contain non-text characters. Each byte of the file is shown as a pair of hex characters. The corresponding text is displayed to the right of each line of hexadecimal data. You can toggle this mode on and off from within LIST with the X key or the heX button on the tool bar. |

You can display spaces rather than periods for non-printable characters by specifying the /XS option. You can also toggle between spaces and periods with the **S** key while displaying a file in hex mode.

4.3.81 LOADBTM

Purpose: Switch a batch file to or from BTM mode

Format: LOADBTM [ON | OFF]

Usage:

TCC recognizes three kinds of [batch files](#): .CMD, .BAT, and .BTM. Batch files with a .BTM extension will run faster than .BAT or .CMD files, as they are loaded into memory at startup and do not open and close the batch file for each line (as do .BAT and .CMD files).

The LOADBTM command turns BTM mode on and off. It can be used to switch modes in a batch file. If you use LOADBTM with no parameter, it will display the current batch mode: LOADBTM ON or LOADBTM OFF.

Using LOADBTM to repeatedly switch modes within a batch file is not efficient. In most cases the speed gained by running some parts of the file in BTM mode will be more than offset by the speed lost through repeated loading of the file each time BTM mode is invoked.

LOADBTM can only be used within a batch file. It is most often used to convert a .BAT or .CMD file to BTM mode without changing its extension.

There is no functional difference between .BAT and .CMD files.

4.3.82 LOADMEDIA

Purpose: Close the door of a removable media drive(s)

Format: LOADMEDIA drive ...

Usage:

LOADMEDIA will close the drive door (if the device allows it) of removable media, such as CD-ROMs, DVDs, etc.

See also [EJECTMEDIA](#).

4.3.83 LOG

Purpose: Save a log of commands to a file

Format: LOG [/A /E /H /W file] [ON | OFF | text]

file The name of the file to hold the log.
text An optional message that will be added to the log.
ON Turns on logging
OFF Turns off logging

[/A\(History Log\)](#)
[/E\(rors\)](#) [/H\(istory log\)](#)
 [/W\(rite to\)](#)

See also: [HISTORY](#).

Usage:

The LOG command provides independent controls for two different methods of logging **TCC** activity:

- [Command Log](#)
- [History Log](#)
- [Output Log](#)

You can only specify one of the /A, /E, and /H options in a single LOG command.

Command Log

Command logging creates a record of each internal and external command executed either from the command prompt or from a batch file in the format below:

[date time] [id] command

where the **date** and **time** are formatted according to the country code set for your system, **id** is the process ID, and **command** is the actual command after any alias or variable expansion.

The default command log filename is **TCCCommandLog**.

History Log

History log creates a record of each command executed from the command prompt exactly as it was

entered, before aliases and variables are expanded, without any additional information.

Output Log

The Output log saves all everything that TCC writes to the console window. It does not log output written by external applications.

Notes

The LOG /H output can be used as the basis for writing batch files. Start LOG /H, then execute the commands that you want the batch file to execute. When you are finished, turn LOG /H off. The resulting file can be turned into a batch file that performs the same commands with little or no editing.

Options:

- /A** This option saves all output to the ***log all*** file. The default filename is ***TCLogAll***.
- /E** This option saves all error messages to the ***error log***. The default filename is ***TCErrorLog***.
See also: the [Error Logging](#) configuration option.
- /H** This option saves the commands to the ***history log***. The default history log name is ***TCHistoryLog***. For example, to turn on history logging and write to the file C:\LOG\HLOG:

```
log /h /w c:\log\hlog
```

- /W** This switch specifies a different filename for the LOG output. It also automatically performs a LOG ON command. For example, to turn command logging on and write the log to C:\LOG\LOGFILE:

```
log /w c:\log\logfile
```

Once you select a new file name with the LOG /W or LOG /H /W command, LOG will use that file until you issue another LOG /W or LOG /H /W command, or until you terminate your **TCC** session. Turning LOG or LOG /H off or on does not change the file name.

4.3.84 LUA

Purpose: Invoke the internal Lua interpreter

Format: LUA [options] [script [args]]

- e _stat_** executes string stat;
- l _mod_** "requires" mod;
- i** enters interactive mode after running script;
- v** prints version information;
- stops handling options;
- executes stdin as a file and stops handling options.

Usage:

After handling its options, lua runs the given script, passing to it the given args as string arguments. When called without arguments, lua behaves as lua -v -i when the standard input (stdin) is the console,

and as lua - otherwise.

Before running any argument, the interpreter checks for an environment variable LUA_INIT. If its format is @_filename_, then lua executes the file. Otherwise, lua executes the string itself.

All options are handled in order, except -i. For instance, an invocation like

```
lua -e'a=1' -e 'print(a)' script.lua
```

will first set a to 1, then print the value of a (which is '=1='), and finally run the file script.lua with no arguments.

Before starting to run the script, lua collects all arguments in the command line in a global table called arg. The script name is stored at index 0, the first argument after the script name goes to index 1, and so on. Any arguments before the script name (that is, the interpreter name plus the options) go to negative indices. For instance, in the call

```
lua -la b.lua t1 t2
```

the interpreter first runs the file a.lua, then creates a table

```
arg = {
  [-2] = "lua",
  [-1] = "-la",
  [0] = "b.lua",
  [1] = "t1", [2] = "t2" }
```

and finally runs the file b.lua. The script is called with arg[1], arg[2], ... as arguments; it can also access these arguments with the vararg expression '=...='.

In interactive mode, if you write an incomplete statement, the interpreter waits for its completion by issuing a different prompt.

If the global variable _PROMPT contains a string, then its value is used as the prompt. Similarly, if the global variable _PROMPT2 contains a string, its value is used as the secondary prompt (issued during incomplete statements). Therefore, both prompts can be changed directly on the command line. For instance,

```
lua -e"_PROMPT='myprompt> '" -i
```

(the outer pair of quotes is for the shell, the inner pair is for Lua), or in any Lua programs by assigning to _PROMPT. Note the use of -i to enter interactive mode; otherwise, the program would just end silently right after the assignment to _PROMPT.

4.3.85 MD / MKDIR

Purpose: Create a subdirectory

Format: MD [/C /D /N[et] /S] path...
or
MKDIR [/C /D /N[et] /S] path...

path The name of one or more directories to create.

[/C\(ompressed\)](#)

[/D \(change directory\)](#)

[/N\(o update\)](#)

[/S\(ubdirectories\)](#)

See also: [RD](#).

Internet: Can be used with [FTP Servers](#).

Usage:

MD and MKDIR are synonyms. You can use either one. If you don't specify any arguments, MD will display its command dialog.

MD creates a subdirectory anywhere in the directory tree. To create a subdirectory from the root, start the **path** with a backslash [\]. For example, this command creates a subdirectory called *MYDIR* in the root directory:

```
md \mydir
```

If no path is given, the new subdirectory is created in the current directory. This example creates a subdirectory called *DIRTWO* in the current directory:

```
md dirtwo
```

To create a directory from the parent of the current directory (that is, to create a sibling of the current directory), start the pathname with two periods and a backslash [..\].

Windows limits the maximum length of the subdirectory name. See [Directories and Subdirectories](#) for details.

When creating a directory on an LFN drive, you must quote any **path** which contains white space or special characters.

If MD creates one or more directories, they will be added automatically to the [extended directory search](#) database unless the /N option is specified.

You can create directories on FTP servers. For example:

```
md ftp://ftp.abc.com/data/index
```

MD sets two internal variables:

%_md_dirs The number of directories created

%_md_errors The number of errors

Options:

/= Display the MD command dialog to help you set the directory and command line options.
You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

/C Create a compressed subdirectory. (Not available in **TCC/LE**.)

- /D** Change to the newly created subdirectory. (Not available in **TCC/LE**.)
- /N** If **/N** has no additional options, do not update the CD / CDD [extended directory search](#) database, *JPSTREE.IDX*. This is useful when creating a temporary directory which you do not want to appear in the extended search database. **/N** takes two optional arguments:
 - e** Don't display errors. (Note that a **/Ne** alone will still update the [extended directory search](#) database.)
 - t** Don't update the [extended directory search](#) database. (This is the same as **/N** with no options.)
- /S** Allows you to create more than one directory at a time. For example, if you need to create the directory *C:\ONE\TWO\THREE* and none of the named directories exist, you can use **/S** to have MD create all of the necessary subdirectories in a single command (without the **/S**, this command will fail because the parent directory *C:\ONE\TWO* does not exist):


```
md /s \one\two\three
```

For compatibility with CMD, **/S** becomes the default if you enable **TCC** extensions with the **/X** switch on the **TCC** startup command line. See [Command Line Options](#) for details on **/X**.

4.3.86 MEMORY

Purpose: Display TCC and Windows memory status

Format: MEMORY

Usage:

MEMORY lists the percentage "memory load" as reported by Windows, the total and available physical RAM, the total and available page file size, the total and available virtual memory, the total and free alias space, the total and free function space (except in **TCC/LE**), and the total history space. The memory load is a figure returned by the operating system which gives an overall sense of memory utilization. It is not a precise indicator of system load or memory usage. The total page file figure shows the total number of bytes that can be stored in the file, but may not reflect the actual size of the current file on disk.

4.3.87 MKLINK

Purpose: Create NTFS symbolic, hard, and soft links

Format: MKLINK [/A /D /H /J /Q /X] Link Target

Link The new symbolic link name

Target The pathname (full or relative) that the new link refers to

- /A** Create a link with an absolute path.
- /D** Create a directory symbolic link. (The default is to create a file symbolic link.)
- /H** Create a hard link (like MKLNK).
- /J** Create a junction.
- /Q** Don't display results.
- /X** Delete directory link.

Usage:

Due to Windows file system restrictions, creating symbolic links with MKLINK requires an NTFS volume and Windows Vista or later.

The file/directory names in **Link** and **Target** can be fully or partially qualified. MKLINK will also copy an existing description to the link. If you don't specify any arguments, MKLINK will display its command dialog.

MKLINK sets two internal variables:

%_mklink_files	The number of links created
%_mklink_errors	The number of errors

See also [MKLNK](#).

Option:

- /=** Display the MKLINK command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A** Create a link with an absolute (full expanded) pathname. For CMD compatibility, MKLINK creates relative links if you don't specify a full pathname.
- /D** Create a directory symbolic link. (The default is to create a file symbolic link.)
- /H** Create a hard link instead of a symbolic link.
- /J** Create a junction rather than a symbolic link.
- /Q** Don't display the result.
- /X** Delete a directory link.

4.3.88 MKLNK

Purpose: Create or delete an NTFS hard or soft link

Format: Create or update a link:

MKLNK [/A:[[-]rhsadecijopt]] parm1 [parm2]

Delete a link

MKLNK /D parm1

parm1 Name of an existing file (hard link) or directory (for soft link).

parm2 Name of the new directory entry (a file or directory reference) to be created.

/A: (Attribute select)

/D Delete a link

See also [MKLINK](#).

File Selection

MKLNK supports the [command dialog](#). For hard links, MKLNK supports [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#). Date, time, size, or file exclusion ranges anywhere on the line apply to all **source** files. Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Usage:

MKLNK is obsolete; you should use [MKLINK](#) for new scripts.

Due to operating and file system restrictions, this command requires an NTFS volume.

The file/directory names in **parm1** and **parm2** can be fully or partially qualified, and may contain wildcards (hard links only). MKLINK will also copy an existing description to the link. If you don't specify any arguments, MKLNK will display its command dialog.

If a single argument is specified and it is a junction, MKLNK will display the directory name linked to the junction.

MKLNK sets two internal variables:

%_mklnk_files	The number of links created
%_mklnk_errors	The number of errors

Hard Links

If **parm1** is a file, and **parm2** does not exist, MKLNK will create a hard link. If **parm2** exists, MKLNK reports an error.

MKLNK (and the underlying Windows API) may fail if the current directory is on a **subst** or **net use** drive, or a **UNC** volume.

Soft Links

If **parm1** is a directory, and **parm2** does not exist, MKLNK will create a soft link, also known as a "directory junction" or "reparse point". If **parm2** exists, and it is a soft link, MKLNK updates it.

A soft link is an indirect or symbolic reference (**parm2**) to a directory that physically resides in another location (**parm1**). Note: deleting files from a soft link is equivalent to deleting the files from the original directory.

Note: Other operating systems, such as Linux, may also support "hard links" and "soft links", but the Windows implementation of these concepts may not behave in the same manner even though the names might be similar.

Option:

/= Display the MKLNK command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

/A: Select only those files that have the specified attribute(s) set (hard links only). See

[Attribute Switches](#) for information on the attributes which can follow **/A:**

- /D Remove an existing hard or soft link. For hard links, if no more links remain **/D** will not delete the file.

4.3.89 MOVE

Purpose: Move files to a new directory (and optionally drive)

Format: MOVE [/A:[[-]rhsadecijopt /B /C /CF /D /E /G /H /I"text" /J /K /L /LD /M /MD /N[dejnst] /O /O:[-]adegnrstu /P /Q /R /S[+]n] /SX /T /U /UF /V /W /Y /Z] [@file] source... destination

source	A file or list of files to move.
destination	The new location for the files.
@file	A text file containing the names of the source files to move, one per line (see @file lists for details).

/A: (Attribute select)	/O (don't move if target exists)
/B (Move after reboot)	/O:... (Order)
/C(hanged)	/P(rompt)
/CF (changed 2s+ resolution)	/Q(uiet)
/D(irectory)	/R(eplace)
/E (No error messages)	/S(ubdirectory tree)
/G (display percent copied)	/SX (single target directory)
/H(idden and system)	/T(total)
/I"text" (match description)	/U(pdate)
/J (copy in restartable mode)	/UF (updated 2s+ resolution)
/K (delete to recycle bin)	/V(erify)
/L (ASCII FTP transfer)	/W(ipe)
/LD (create link)	/Y (force move of encrypted files)
/M(odified files)	/Z (overwrite)
/MD (Create target directory)	
/N (Disable)	

See also [COPY](#), [DEL](#) and [RENAME](#).

File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), [delayed variable expansion](#), and [include lists](#). Date, time, size, or file exclusion ranges anywhere on the line apply to all **source** files. Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Internet: Can be used with [FTP/TFTP/HTTP/HTTPS Servers](#).

Usage:

The MOVE command moves one or more files from one directory to another, whether the directories are on the same drive or not. It has the same effect as copying the files to a new location and then deleting the originals. Like [COPY](#) and [RENAME](#), MOVE works with single files, multiple files, and sets of files specified with an include list. If you don't specify any arguments, MOVE will display its command dialog.

The simplest MOVE command moves a single **source** file to a new location and, optionally, gives it a new name. These two examples both move one file from drive C: to the root directory on drive A:

```
[c:\] move myfile.dat a:\  
[c:\] move myfile.dat a:\savefile.dat
```

In both cases, *MYFILE.DAT* is removed from drive C: after it has been copied to drive A:. If a file called *MYFILE.DAT* in the first example, or *SAVEFILE.DAT* in the second example, already existed on drive A:, it would be overwritten. (This demonstrates the difference between MOVE and RENAME. MOVE will move files between drives and will overwrite the destination file if it exists; RENAME will not.)

When you move a single file, the **destination** can be a directory name or a file name. If it is a directory name, and you add a backslash [\\] to the end of the name, MOVE will display an error message if the name does not refer to an existing directory. You can use this feature to keep MOVE from treating a mistyped **destination** directory name as a file name, and attempting to move the **source** file to that name.

If you MOVE multiple files, the **destination** must be a directory name. MOVE will move each file into the **destination** directory with its original name. If the **destination** is not a directory, MOVE will display an error message and exit. For example, if C:\FINANCE\MYFILES is not a directory, this command will display an error; otherwise, the files will be moved to that directory:

```
move *.wks *.txt c:\finance\myfiles
```

The /D option can be used for single or multiple file moves; it checks to see whether the **destination** is a directory, and will prompt to see if you want to create the **destination** directory if it doesn't exist.

If MOVE creates one or more destination directories, they will be added automatically to the extended directory search database; see [Extended Directory Searches](#) for details.

Be careful when you use MOVE with the SELECT command. If you SELECT multiple files and the **destination** is not a directory (for example, because of a misspelling), MOVE will assume it is a file name. In this case each file will be moved in turn to the **destination** file, overwriting the previous file, and then the original will be erased before the next file is moved. At the end of the command, all of the original files will have been erased and only the last file will exist as the **destination** file.

You can avoid this problem by using square brackets with SELECT instead of parentheses (be sure that you don't allow the command line to get too long; watch the character count in the upper left corner while you're selecting files). MOVE will then receive one list of files to move instead of a series of individual filenames, and it will detect the error and halt. You can also add a backslash [\\] to the end of the **destination** name to ensure that it is the name of a subdirectory (see above).

When you specify a single subdirectory source and a single subdirectory target, the source directory tree will be moved to a subdirectory of the target directory. If the source is a subdirectory and the target doesn't exist, the target subdirectory will be created and the source tree moved to it. (These are both for compatibility with CMD.)

If you specify the /C, /CF, /R, /U, or /UF options, MOVE will append a ! to the move specifier if the target exists and is being overwritten. For example:

```
[d:]      move file1 file2  
file1 ->! file2
```

MOVE sets three internal variables:

%_move_dirs	The number of directories created
%_move_files	The number of files moved
%_move_errors	The number of errors

- **FTP Usage:**

You can move files to and from Internet URLs (FTP, TFTP and HTTP). For example:

```
move ftp://ftp.abc.com/f1.txt c:\text\
```

Files moved to or from FTP servers are normally transferred in binary mode. To perform an ASCII transfer use the **/L** switch. File descriptions are not copied when moving files to an Internet URL.

Wildcard characters such as [*] and [?] will be treated as wildcards in FTP URLs, but will be treated as normal characters in HTTP URLs.

Note: The **/G** option (percent moved) may report erratic values during transfer of files larger than 4 Gb (an FTP limitation) and during http downloads.

- **NTFS File Streams:**

MOVE supports file streams on NTFS drives. You can move an individual stream by specifying the stream name, for example:

```
move streamfile:s1 file2
```

If no stream name is specified the entire file is moved, including all streams. However, if you move a file to a drive or device which does not support streams, only the file's primary data is moved; any additional streams are not processed and their data will be lost.

See [NTFS File Streams](#) for additional details.

- **Advanced Features and Options**

If MOVE must physically copy the files and delete the originals (rather than renaming them), then some disk space may be freed on the **source** drive. The free space may be the result of moving the files to another drive, or of overwriting a larger **destination** file with a smaller **source** file. MOVE displays the amount of disk space recovered unless the **/Q** option is used (see below). It does so by comparing the amount of free disk space before and after the MOVE command is executed. However, this amount may be incorrect if you are using a deletion tracking system which retains deleted files for later recovery, or if another program performs a file operation while the MOVE command is executing.

Use caution with the **/A:** and **/H** switches (both of which can allow MOVE to process hidden files) when you are physically moving files, and both the **source** and **destination** directories contain file descriptions. If the **source** file specification matches the description file name (normally **DESCRIPT.ION**), and you tell MOVE to process hidden files, the **DESCRIPT.ION** file itself will be moved, overwriting any existing file descriptions in the **destination** directory. For example, if the C:\DATA directory contains file descriptions, this command would overwrite any existing descriptions in the D:\SAVE directory:

```
[c:\data] move /h d* d:\save\
```

(If you remove the hidden attribute from the **DESCRIPT.ION** file the same caution applies even if you do

not use **/A:** or **/H**, as *DESCRIPTION* is then treated like any other file.)

Note: The wildcard expansion process will attempt to allow both CMD-style "extension" matching (only one extension, at the end of the word) and the advanced **TCC** string matching (allowing things like ***.*.abc**) when an asterisk is encountered in the **destination** of a MOVE command.

MOVE supports [regular expression](#) back references in the target name. If you are using back references, you must also use a regular expression in the source name. The syntax is:

```
move ::filename ::target
```

MOVE supports connected web folders. If an HTML file (i.e., with an .htm or .html extension) is copied, MOVE will look for a folder in the same directory with the same name and an extension of ".files". If it is found, the .files directory will be moved to the target directory. You can disable connected web folders by setting the registry key:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\NoFileFolderConnection=0
```

Options:

- /=** Display the MOVE command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A:** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow **/A:**. See the cautionary note under **Advanced Features and Options** above before using **/A:** when both the **source** and **destination** directories contain file descriptions. Do not use **/A:** with @file lists. See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /B** If MOVE can't move the file (i.e., access denied), it will schedule it to be moved at the next reboot.
- /C** Move files only if the **destination** file exists and is older than the **source** (see also **/U**). This option is useful for updating the files in one directory from those in another without moving any newly-created files. Do not use **/C** with @file lists. See [@file lists](#) for details.
- /CF** Move files only if the **destination** file exists and is more than 2 seconds older than the **source** (see also **/U** and **/UF**). Do not use **/CF** with @file lists. See [@file lists](#) for details. (Not available in **TCC/LE**.)
- /D** Requires that the **destination** be a directory. If the **destination** does not exist, MOVE will prompt to see if you want to create it. If the **destination** exists as a file, MOVE will fail with an "Access denied" error. Use this option to avoid having MOVE accidentally interpret your **destination** name as a file name when it's really a mistyped directory name.
- /E** Suppress all non-fatal error messages, such as "File Not Found." Fatal error messages, such as "Drive not ready," will still be displayed. This option is most useful in batch files and aliases.

- /G** Displays the percentage of the file moved, the transfer rate (in Kbytes/second), and the estimated time remaining. This is useful when copying large files across networks or via FTP to show whether the move is proceeding. /G will also display the % moved even if Windows is doing a rename (which may be a copy & delete internally).
- /H** Move all files, including hidden and system files. See the cautionary note under **Advanced Features and Options** above before using /H when both **source** and **destination** directories contain file descriptions.
- /I"text"** Select **source** files by matching text in their descriptions. The text can include wildcards. The search text must be enclosed in double quotes, and must follow the /I immediately, with no intervening spaces. You can select all filenames that have a description with /I"[?]*", or all filenames that do not have a description with /I"\"". Do not use /I with @file lists. See [@file lists](#) for details.
- /J** Copy the file in restartable mode. The copy progress is tracked in the destination file in case the move fails. The copy can be restarted by specifying the same source and destination file names.
- /K** If the MOVE is to a different drive, move the source file to the recycle bin instead of deleting it.
- /L** Perform FTP transfers in ASCII mode, instead of the default binary mode.
- /LD** When used with /S, if the source is a symbolic or hard link to a directory, MOVE will create the link in the target directory instead of moving the subdirectory tree.
- /M** Move only files that have the archive bit set. The archive bit will remain set after the MOVE. Do not use /M with @file lists. See [@file lists](#) for details.
- /MD** Create the target directory if it doesn't exist. (Note that you *must* either terminate the target directory name with a trailing \ or specify a filename component; otherwise MOVE cannot tell what you want for the directory and what you want for the filename!)
- /N** Do everything except actually move the file(s). This option is most useful for testing what a complex MOVE command will do. /N displays how many files would be moved. /N does not prevent creation of **destination** subdirectories when it is used with /S.

A /N with one or more of the following arguments has an alternate meaning:

- d** Skip hidden directories (when used with /S)
- e** Don't display errors.
- j** Skip junctions (when used with /S)
- n** Don't update the file descriptions
- s** Don't display the summary.
- t** Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*).

- /O** Don't move the file(s) unless the target doesn't exist, i.e. do not overwrite an existing target..
- /O:....** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the

listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n Sort by filename and extension, unless e is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d Sort by date and time (oldest first); also see **/T:acw**
- e Sort by extension
- g Group subdirectories first, then files
- r Reverse the sort order for all options
- s Sort by size
- t Same as d
- u Unsorted

- /P** Prompt the user to confirm each move. Your options at the prompt are explained in detail under [Prompts](#).
- /Q** Don't display filenames, the total number of files moved, the percentage moved, or the amount of disk space recovered, if any. When used in combination with the **/P** option above, it will prompt for filenames but will not display the totals. This option is most often used in batch files. See also **/T**.
- /R** Prompt for a **Y** or **N** response before overwriting an existing **destination** file.
- /S** Move an entire subdirectory tree to another location. MOVE will attempt to create the **destination** directories if they don't exist, and will remove empty subdirectories after the move. When **/D** is used with **/S**, you will be prompted if the first **destination** directory does not exist, but subdirectories below that will be created automatically by MOVE. If MOVE /S creates one or more destination directories, they will be added automatically to the **JPSTREE.IDX** database. If you attempt to use **/S** to move a subdirectory tree into part of itself, MOVE will detect the resulting infinite loop, display an error message, and exit. You cannot combine multiple **/S** options (including **/S**, **/Sn**, **/S+1**, or **/SX**) in a single command, or use any **/S** option with @file lists. See [@file lists](#) for details.

If you specify a number after the **/S**, MOVE will limit the subdirectory recursion to that number. For example, if you have a directory tree "**\a\b\c\d\e**", **/S2** will only affect the "a", "b", and "c" directories.

If you specify a **+** followed by a number after the **/S**, MOVE will not move any files until it gets to that depth in the subdirectory tree. For example, if you have a directory tree **\a\b\c\d\e**, **/S+2** will not move anything in **\a** or **\a\b**. (Not available in **TCC/LE**.)

MOVE will display the empty source subdirectories it is removing (unless you use the **/Q** option).

- /SX** Move the subdirectory tree to a single target directory (implies **/S**). (Not available in **TCC/LE**.) MOVE will remove empty subdirectories after the move. You cannot combine multiple **/S** options (including **/S**, **/Sn**, **/S+1**, or **/SX**) in a single command, or use any **/S** option with @file lists. For example, to move all of the .EXE files in **c:\files** and all of its subdirectories to the directory **d:\execfiles**:

```
copy /sx c:\files\*.exe d:\lexefiles\
```

- /T Don't display filenames as they are moved, but display the total number of files moved.
- /U Move each **source** file only if it is newer than a matching **destination** file or if a matching **destination** file does not exist (also see /C). This option is useful for moving new or changed files from one directory to another. Do not use /U with @file lists. See [@file lists](#) for details. When used with file systems that have different time resolutions (such as FAT and NTFS), /U will attempt to use the "coarsest" resolution of the two.
- /UF Move each **source** file only if it is more than 2 seconds newer than a matching **destination** file or if a matching **destination** file does not exist (also see /C and /CF). Do not use /UF with @file lists. See [@file lists](#) for details. (Not available in **TCC/LE**.)
- /V Verify each disk write by performing a true byte-by-byte comparison between the source and the newly-created target file. This option may significantly increase the time necessary to complete a MOVE command. /V will not work for FTP, TFTP, or HTTP moves.
- /W If the MOVE is to a different drive, after the move overwrite the source file contents using the DoD 5220.22-M (E) standard for secure deletion. (This overwrites every byte in the file three times with different values). Use this option to completely obliterate a file's contents from your disk. Once you have used this option it is impossible to recover the file even if you are using an undelete utility, because the contents of the file are destroyed before it is deleted.
- /Y Force copy of an encrypted file even when the target will be decrypted (for CMD compatibility).
- /Z Overwrite read-only destination files. Without this option, MOVE will fail with an "Access denied" error if the destination file has its read-only attribute set. This option allows MOVE to overwrite read-only files without generating any errors.

4.3.90 MSGBOX

Purpose: Display a Windows message box

Format: MSGBOX [/1["text"] /2["text"] /3["text"] /4["text"] /Dn /H /I /L /M /N /O /Px,y /Q /R /S /Tn /V /W] buttonstype ["title"] prompt

buttonstype	One of OK , OKCANCEL , YESNO , YESNOCANCEL , RETRYCANCEL , ABORTTRYIGNORE , CANCELTRYCONTINUE , or CONTINUEABORT
title	Text for the title bar of the message box.
prompt	Text that will appear inside the message box.

/1 (1st button)	/N (no sound)
/2 (2nd button)	/O (topmost window)
/3 (3rd button)	/P (screen coordinates)
/4 (4th button)	/Q(uestion icon)
/D(isable temporarily)	/R(ight justify buttons)
/H(elp button)	/S(top icon)
/I(nformation icon)	/T(imeout)
/L(imit width)	/V(ista style)

[/M \(system modal\)](#)[/W\(arning icon\)](#)

See also: [INKEY](#), [INPUT](#), [QUERYBOX](#), and [TASKDIALOG](#).

Usage:

MSGBOX can display one of eight kinds of message boxes and wait for the user's response. You can use **title** and **prompt** to display any text you wish. TCC will automatically size and center the message box on the screen. The message box has up to three response buttons (plus an optional Help button), depending on its type, as shown below.

button type	button 1	button 2	button 3
OK	OK		
OKCANCEL	OK	Cancel	
YESNO	Yes	No	
YESNOCANCEL	Yes	No	Cancel
TRYCANCEL	Retry	Cancel	
ABORTTRYIGNORE	Abort	Retry	Ignore
CANCELTRYCONTINUE	Cancel	Try Again	Continue
CONTINUEABORT	Continue	Abort	

If the standard message box types don't meet your needs, you can create a custom message box with up to four buttons (plus an optional Help button), specifying the text that appears on each button.

The button the user chooses is indicated using the internal variable [% ?](#). Be sure to save the return value in another variable or test it immediately; because the value of [% ?](#) changes with every internal command. The following list shows the value returned for each selection:

response	% ?
Yes or OK	10
No	11
Cancel	12
Retry	13
Try Again	14
Continue	15
Ignore	16
Abort	17
Help	18
timeout	20
custom button 1	21
custom button 2	22
custom button 3	23
custom button 4	24

If you define custom buttons, the button type argument will be ignored.

If there is an error in the MSGBOX command itself, [% ?](#) will be set as described in its documentation (see [?](#)).

For example, to display a Yes or No message box and take action depending on the result, you could use commands like this:

```
msgbox yesno "Copy" Copy all files to A:?
if %_? == 10 copy * a:
```

Since MSGBOX doesn't write to standard output, it disables redirection and piping to allow you to enter the redirection characters (<, >, and |) in your prompt text.

MSGBOX creates a popup dialog box. If you prefer to retrieve input from the command line, see the [INKEY](#) and [INPUT](#) commands.

Options:

- /1 If there is a text string following the option, set the custom text for the first button.
Otherwise, set the first button as the default.
- /2 If there is a text string following the option, set the custom text for the second button.
Otherwise, set the second button as the default.
- /3 If there is a text string following the option, set the custom text for the third button.
Otherwise, set the third button as the default.
- /4 If there is a text string following the option, set the custom text for the fourth button.
Otherwise, set the fourth button as the default.
- /Dn Disable the message box buttons for *n* seconds at startup.
- /H Display a help button.
- /I Display an icon consisting of a lower case "i" in a circle in the message box.
- /L Limit the maximum message box width to no more than 1/3 the screen width (unless the button text requires more). (Not available in **TCC/LE**.)
- /M The message box window will be displayed on top of all other windows.
- /N Don't play the default sound.
- /O The message box is created as a topmost window.
- /Px,y The initial x,y screen coordinates. If you don't use this option, MSGBOX will center its window in the **TCC** tab window.
- /Q Display a question mark icon in the message box.
- /R The buttons will be right-justified (as in XP Explorer).
- /S Display a stop sign icon in the message box.
- /Tn MSGBOX will wait a maximum of *n* seconds for a response (and then close). If the time limit expires, %_? will be set to 20. The time remaining before the window closes will be displayed in the default button.
- /V Display the message box in the Windows 7 / 8 style (the message background will be the

current window color, the buttons will be right-justified and slightly bigger, and the position of icon and message will be adjusted.) This is the default in Vista and later. (Not available in **TCC/LE**.)

/W Display an exclamation point icon in the message box.

4.3.91 NETMONITOR

Purpose: Monitor network connection and disconnection

Format: NETMONITOR [/C [name]]
NETMONITOR name CONNECTED | DISCONNECTED n command

name Network name

n Number of repetitions (or **FOREVER**)

command Command to execute when condition is triggered

[/C\(lear\)](#)

Usage:

The network name can be either **LAN** (for a local area network), **WAN** (dialup network), or the name of a wireless network. The network name can include wildcards.

The command line will be parsed and expanded before NETMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. NETMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

If you don't enter any arguments, NETMONITOR will display the networks it is currently monitoring.

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

NETMONITOR creates environment variables when a network is connected that can be queried by **command**. The variable is deleted after **command** is executed.

_netname The name (SSID) of the network

_netcount The number of times the condition has been triggered

Options:

/C If **name** is specified, remove the monitor for that network. Otherwise, remove all network monitors.

4.3.92 ON

Purpose: Execute a command in a batch file when a specific condition occurs

Format:

- ON BREAK [command]
- ON CLOSE [command]
- ON CONDITION [condition command]
- ON DBLCLICK [command]
- ON ERROR [command]
- ON ERRORLEVEL n [command]
- ON ERRORMSG [command]
- ON LOGOFF [command]
- ON LBUTTON [command]
- ON MBUTTON [command]
- ON RBUTTON [command]
- ON RESUME [command]
- ON SHUTDOWN [command]
- ON SUSPEND [command]

command command to execute when the event occurs

Usage:

ON sets a watch that remains in effect for the duration of the current batch file, or until replaced by another ON command of the same type. Whenever a **break** or **error** condition occurs after ON has been executed, the corresponding **command** is automatically executed. You can have multiple ON commands active at a time, as long as no two are the same type. (For example, you can have an ON BREAK and an ON CLOSE, but not two ON LBUTTON.)

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. ON will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

Activation of ON BREAK

ON BREAK will execute **command** if the user presses **Ctrl-C** or **Ctrl-Break**.

Activation of ON CLOSE

ON CLOSE will execute **command** when the **TCC** tab is closed.

Activation of ON CONDITION

ON CONDITION will execute **command** when **condition** is true. **condition** can be any test that is valid in **IF**. The test will be done after each command is executed. If you are executing a loop (**DO** or **FOR**), the test will be done each time through the loop. (Not available in **TCC/LE**.)

Activation of ON DBLCLICK

ON DBLCLICK will execute **command** when the left mouse button is double clicked when **TCC** is the active window. (Note that if you also have an ON LBUTTON command, it will be executed on the first click.)

Activation of ON ERROR and ON ERRORMSG

ON ERROR or ON ERRORMSG will execute ***command*** after any critical error, operating system error (such as a disk write error) or internal command error (such as a [COPY](#) command that fails to copy any files, or the use of an invalid command option).

ON ERROR executes ***command*** immediately after the error occurs, without displaying any **TCC** error message (Windows errors may still be displayed).

ON ERRORMSG first displays the appropriate error message, then executes ***command***.

If both are specified, ON ERROR will take precedence, and ON ERRORMSG will be ignored.

Activation of ON ERRORLEVEL

ON ERRORLEVEL *n* will execute ***command*** when the internal ERRORLEVEL variable is equal to the integer specified by ***n***. You can also use the IF ERRORLEVEL tests; for example:

```
ON ERRORLEVEL EQ 37 ...
```

Activation of ON LBUTTON

ON LBUTTON will execute ***command*** when the left mouse button is clicked.

Activation of ON LOGOFF

ON LOGOFF will execute ***command*** when the user logs off.

Activation of ON MBUTTON

ON MBUTTON will execute ***command*** when the middle mouse button is clicked when **TCC** is the active window.

Activation of ON RBUTTON

ON RBUTTON will execute ***command*** when the right mouse button is clicked when **TCC** is the active window.

Activation of ON RESUME

ON RESUME will execute ***command*** when the system resumes after sleeping or hibernating. (Not available in **TCC/LE**.)

Activation of ON SHUTDOWN

ON SHUTDOWN will execute ***command*** when the system is being shut down.

Activation of ON SUSPEND

ON SUSPEND will execute ***command*** when the system is going to sleep or hibernation. In Vista or later, the system will continue suspending after a maximum of 2 seconds. (Not available in **TCC/LE**.)

Scope

Each time an ON statement is defined, it defines a new command to be executed for that event, and any prior command is discarded.

If you do not specify a command, **TCC** restores the default handler.

An ON statement only affects the current batch file. When the batch file containing ON is exited for any reason, whether temporarily (e.g., by a [CALL](#) to another batch file) or permanently, the **TCC** default **break** and **error** handlers become effective. A [CALLED](#) batch file may then use ON to define its own handlers. When control returns to the calling batch file, its **break** and **error** handlers that had been in effect at the [CALL](#) are reactivated.

Operation

The command can be any command that can be used on a batch file line by itself. Frequently, it is a [GOTO](#) or [GOSUB](#) command. For example, the following fragment traps any user attempt to end the batch file by pressing **Ctrl-C** or **Ctrl-Break**. It scolds the user for trying to end the batch file and then continues:

```
on break gosub gotabreak
do i = 1 to 1000
    echo %i
enddo
quit
:gotabreak
echo Hey! Stop that!!
return
```

You can use a [command group](#) as the command if you want to execute multiple commands, for example:

```
on break (echo Oops, got a break! & quit)
```

ON assumes that you want to continue executing the batch file. After the command is executed, control automatically returns to the command in the batch file immediately after the one that was interrupted by the event. To avoid continuing the batch file after the event at the next command perform one of the following in **command**:

- transfer control with [GOTO](#),
- end the batch file with [QUIT](#) or [CANCEL](#)
- chain to another batch file (without using [CALL](#)).

When handling an error condition with ON ERROR[MSG], you may find it useful to use [internal variables](#), particularly [% ?](#) and [% SYSERR](#), to help determine the cause of the error.

To force **TCC** to ignore break or error, use the [REM](#) command as your command.

Limitations

ON can only be used in batch files.

The ON ERROR[MSG] command will not be invoked if an error occurs while reading or writing redirected input, output, or a pipe.

Caution: If a break or error occurs while the command specified in ON BREAK, ON ERROR, ON ERRORLEVEL, or ON ERRORMSG is executing, the command will be restarted. This means you must use caution either to avoid or to handle any possible errors in the commands invoked by ON, since such errors can cause an infinite loop.

4.3.93 OPTION

Purpose: Modify or display **TCC** configuration

Formats: [Invoking the OPTION dialog:](#)

OPTION

[Check for updates:](#)

OPTION /U

[Temporarily changing an option:](#)

OPTION //directive=value ...

[Temporarily changing a list of options:](#)

OPTION @filename

[Displaying the current value of an option:](#)

OPTION directive

directive Name of a directive to set, modify, or display.

value A new value for that directive.

filename A file containing directives to be immediately activated.

See also: [.INI file](#), [SETDOS](#)

Usage:

Invoking the OPTION Dialog

OPTION without parameters displays a property sheet which allows you to modify most of the configuration options stored in the [INI file](#).

When you exit from the property sheet, you can select **Save** to save your changes in the .INI file for use in the current session and all future sessions, or select **Cancel** to discard the changes. See [Configuration Dialogs](#) for more information.

In some cases, changes you make in the **Startup** section of the OPTION dialogs will only take effect when you restart **TCC**. Other changes take effect as soon as you exit the dialogs with **Apply** or **OK**. However, not all option changes will appear immediately, even if they have taken effect. For example, some color changes will only appear after a [CLS](#) command.

OPTION handles most standard directives. The [Key Mapping Directives](#) and [Advanced Directives](#) cannot be modified with the OPTION dialogs. These settings must be manually edited .INI file.

OPTION does not preserve inline comments when saving modified settings in the .INI file. To be sure .INI file comments are preserved, put them on separate lines in the file.

Check for Updates

The /U option will invoke the updater to check <http://jpsoft.com> for updates to ***Take Command / TCC***.

Setting Individual Options Temporarily

If you follow the OPTION command with one or more sequences of a double slash mark //, each followed by a new ***directive=value***, the new settings will take effect immediately, and will be in effect for the current session only. This example turns off batch file echo and changes the input colors to bright cyan on black:

```
option //BatchEcho=No //InputColors=bri cya on bla
```

Option values may contain white space. However, you cannot enter an option value that contains the // string. If you do not specify a value, OPTION will reset the value for that directive to the default.

This feature is most useful for testing settings quickly, and in aliases or batch files that depend on certain options being in effect.

Changes made with // are temporary. They will not be saved in the .INI file.

Setting Many Options Temporarily

The command OPTION @***filename*** allows you to temporarily modify multiple directive settings. The file specified by ***filename*** must be in the same format as an [.INI file](#). Changes made with @***filename*** are temporary. They will not be saved in the .INI file.

Displaying an option value

Specifying an option name alone will display the value of that option; e.g.:

```
option localHistory
localHistory=Yes
```

See also: the [@OPTION](#) function.

4.3.94 OSD

Purpose: Write floating text to the display

Format: OSD [/ID=n /C[n] /Font=n /ID=n /N /POS=top, left /RGB=r,g,b /TIME=n /TOP /BOTTOM /LEFT /RIGHT /HCENTER /VCENTER /V] text

/ID=n	Open the OSD window n (0-9). /ID is optional; it will default to 0. If /ID is specified, it must be the first argument.
/C[n]	Close the specified OSD display. /C=n must be the only argument. /C will default to OSD window 0.
/Font=n	The font height (default 18)
/N	Don't wait for timeout before returning to the prompt
/POS=top, left	Screen coordinates for the top left corner of the text (default 10,10)
/RGB=r,g,b	Text color in RGB format (default 0,255,0)
/TIME=n	Time in seconds to display the text (default 10)
/TOP	Position the text at the top of the display
/BOTTOM	Position the text at the bottom of the display

/LEFT	Position the text at the left of the display
/RIGHT	Position the text at the right of the display
/HCENTER	Center the text horizontally
/VCENTER	Center the text vertically
/V	Display the text vertically
text	The text to display

Usage:

OSD displays text on the desktop without a surrounding window, like TV or monitor prompts.

If you want to display multiple lines, insert the LF escape sequence (^N) in your text. For example:

```
osd /pos=40,50 This is text with^Nmultiple lines.
```

If you specify the /V (vertical display) option, you cannot also display multiple lines of text.

You can combine the window positioning options. For example:

```
osd /hcenter /vcenter /n Your text here
```

OSD will strip leading whitespace in **text**.

You can control up to 10 simultaneous OSD windows with the /ID=n and /C=n options. If you don't specify /ID, OSD will default to window 0.

4.3.95 PATH

Purpose: Display or alter the list of directories that **TCC** will search for executable files, batch files, and files with executable extensions that are not in the current directory

Format: PATH [directory [;directory...]]

directory The full name of a directory to include in the path setting.

See also: [ESET](#) and [SET](#) (the PATH command is syntactically equivalent to SET PATH).

Usage:

When **TCC** is asked to execute an external command (an *.EXE*, *.BTM*, *.BAT*, or *.CMD* file, or an executable extension), it first looks for the file in the current directory. If it fails to find an executable file in the current directory, it will search each of the directories specified in the PATH setting.

TCC first searches the current directory before any directories listed in your search path. For example, after the following PATH command, **TCC** will search for an executable file in four directories: the current directory, the root directory on drive C, then the *BIN* subdirectory on C, and then the *UTIL* subdirectory on C:

```
path c:\;c:\bin;c:\util
```

The list of **directories** to search is stored as an environment string, and can also be set or viewed with [SET](#), and edited with [ESET](#).

The [PATHEXT](#) environment variable, and the related [PathExt](#) configuration option, can be used to select

the extensions to look for when searching the PATH for an executable file.

If you enter PATH with no parameters, the current path is displayed:

```
[c:\] path  
PATH=C:\;C:\BIN;C:\UTIL
```

Entering PATH and a semicolon clears the search path so that only the current directory is searched for executable files. Some applications also use the PATH to search for their files.

If you include an explicit file extension on a command name (for example, WP.EXE), the search will find files with that name and extension in the current directory and every directory in the path. It will not locate other executable files with the same base name (i.e., WP.CMD).

If you have an entry in the path which consists of a single period [.], the current directory will not be searched first, but instead will be searched when **TCC** reaches the "." in the path. This allows you to delay the search of the current directory for executable files and files with executable extensions. In rare cases, this feature may not be compatible with applications which use the path to find their files; if you experience a problem, you will have to remove the "." from the path while using any such application.

If you specify an invalid directory in the path, it will be skipped and the search will continue with the next directory in the path.

4.3.96 PAUSE

Purpose: Suspend batch file or alias execution

Format: PAUSE [text]

text The message to be displayed as a user prompt.

Usage:

A PAUSE command will suspend execution of a batch file or alias, giving you the opportunity to change disks, turn on the printer, etc.

PAUSE waits for any key to be pressed and then continues execution. You can specify the **text** that PAUSE displays while it waits for a keystroke, or let it use the default message:

```
Press any key when ready...
```

For example, the following batch file fragment prompts the user before erasing files:

```
pause Press Ctrl-C to abort, any other key to erase all .LST files  
erase *.lst
```

If you press **Ctrl-C** or **Ctrl-Break** while PAUSE is waiting for a key, execution of an alias will be terminated, and execution of a batch file will be suspended while you are asked whether to cancel the batch job. In a batch file, you can handle **Ctrl-C** and **Ctrl-Break** yourself with the [ON BREAK](#) command.

4.3.97 PDIR

Purpose: Display information about files and subdirectories in user-definable fields. It is a "programmable DIR" command.

Format: PDIR [ranges] [/A:[attrlist] /B /D /H /I"text" /K /M /N[dej] /O:[order] /P /S[[+]n] /T:t /(...)] [file...]

attrlist	Selection attributes (see attribute switches for details)
order	Hierarchical list of sort keys
ranges	One or more date, description, exclusion, size, time ranges
file	One or more files to list
t	Timestamp type selection code

/A:	Attribute select	/N	Disable options
/B	Bare filenames	/O	Order
/D	colorize	/P	Page pause
/H	do not Hide . and ..	/S	Subdirectories
/I"text"	description range	/T[:t]	Timestamp type
/K	show header	/(...)	output fields and format
/M	show footer		

See also: [DIR](#), [ATTRIB](#), [DESCRIBE](#), and [SELECT](#).

File Selection

Supports [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

Internet

Can be used with [FTP/HTTP Servers](#).

Usage

PDIR is an extremely flexible command allowing you to display information about files and directories from one or more local or remote volume or directories in a wide array of user-defined formats. For a simpler version, see the [DIR](#) command.

PDIR and [DIR](#) are related, but they do not have identical switches and they are not intended to produce identical output. PDIR is primarily intended to produce output that will be subsequently parsed by another program (or batch file), or (more rarely) for a special-purpose directory display. Its options and output are geared towards those applications.

The various PDIR displays are controlled through options or switches. The best way to learn how to use the many options available with the PDIR command is to experiment. You will soon know which options you want to use regularly. You can then select those options permanently by using the [ALIAS](#) command.

The [/\(...\)](#) option specifies which fields you want to display and how to format them. (You can have multiple [/\(...\)](#) options on a line.) The syntax is:

- a** Attributes
- c** Compression: Display the compression percentage on NTFS drives with compression enabled.
- d[...]** Date (you must specify at least one subfield, otherwise the field remains blank)

d day (2 digits, leading zero)
m month (2 digits, leading zero)
y year (4 digits)

f[...] File or Directory name (case sensitive)

P SFN path
p LFN path
N SFN filename
n LFN filename (default)
q Enclose the filename in double quotes if it contains whitespace or special characters (not available in **TCC/LE**)

i Description

m MD5 hash value (see the [@MD5](#) function)

q File or directory owner (NTFS only)

r CRC32 hash value (see the [@CRC32](#) function)

s stream names (NTFS only)

sp path and stream names as pathname+filename+streamname (NTFS only)

t[...] Time (you must specify at least one subfield, otherwise the field remains blank)

h hours (2 digits, leading zero)
m minutes (2 digits, leading zero)
s seconds (2 digits, leading zero)
d milliseconds (decimal separator and 3 digits)

z[...] Size

a Allocated size (this will usually be more than the physical size unless the file is compressed.) Note that you cannot get the allocated size on FTP servers or network sharenames.
c The size will be formatted using the thousands separator (default is a comma)
k|K|m|M|g|G|t|T (case sensitive) format as kilobytes, megabytes, gigabytes, or terabytes, as used in variable functions (see [Memory Size / Disk Space / File Size Units and Report Format](#)). Note that the size will be truncated, not rounded.

@function[*]

call the specified variable [function](#) (internal or user-defined). To specify the current filename, use * as the parameter. For example, **pdir / (f @md5 [*])** displays the filename and the MD5 hash. Note that the % prefix of the function name is NOT used with the symbolic * parameter. If the parameter of the function is not the symbolic * or it is an "inner" function the % prefix must be doubled, e.g., **@function1[%%@function2[*]]**

" . . ." Literal string (in quotes). Characters are displayed as is, except that escape characters are converted.

You can also specify a format, independently for each field, by prefixing the field character with its format

specification:

`[-]i.a`

where

- specifies left justification instead of the default, right justification;
- i specifies the minimum field width, and
- a specifies the maximum field width.

If the first digit of *i* is **0**, the field will be padded with zeros instead of spaces. Some fields cannot be reduced below a minimum width (for example, the **z** (size) field is a minimum of 15 digits).

If a PDIR line is empty (for example, if you have an embedded @IF), it will not be displayed.

If you want to append fields with no intervening whitespace, or with a custom delimiter character, you can use double quotes to specify arguments. For example, to display the date and time with no space between them:

`pdir /(dymd""thms) *`

Or to display the date and time separated by a +:

`pdir /(dymd"+thms) *`

PDIR sets three internal variables:

<code>%_pdir_dirs</code>	The number of directories created
<code>%_pdir_files</code>	The number of files moved
<code>%_pdir_errors</code>	The number of errors

Example

To display the CRC, the full LFN and the owner of each file:

`pdir / (r fpn q) *`

Options

Options on the command line apply only to the filenames which follow the option, and options at the end of the line apply to the preceding filename only. This allows you to specify different options for different groups of files, yet retains compatibility with the traditional [DIR](#) command when a single filename is specified.

Most options are used to select the desired files/directories. (This is in contrast to the [DIR](#) command.) The special option [/\(...\)](#) is used to specify which characteristics of the selected files or directories should be displayed in which sequence and format.

/A:... . Display only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow **/A:.**

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

- /B** Suppress the header and summary lines, and display file or subdirectory names only, in a single column. This option is most useful when you want to redirect a list of names to a file or another program. If you use **/B** with **/S**, PDIR will show the full path of each file (the same display as **/F**) instead of simply its name and extension. If you use **/B** with **/X** on an LFN drive, PDIR will display the short name of each file instead of the long name. **/B** also sets **/H**.
- /B1** will display relative paths when used with **/S**. (Normally, **/B** shows the full pathname for the file.)
- /D** Colorize the directory listing. See [DIR](#) for more information on directory colorization.
- /H** Show the "." and ".." directory names (normally suppressed).
- /I"text"** Select filenames by matching text in their descriptions. See [Description Ranges](#) for details.
- /K** Show the header (disk and directory name) display.
- /M** Show the footer (file and byte count totals) display.
- /N** Turn off the specified options.
- d** Skip hidden directories (when used with **/S**)
 - e** Don't display errors
 - j** Skip junctions (when used with **/S**)
 - m:n** Display a maximum of *n* directory entries
- /O...** The sorting order is applied to the listings of each subdirectory separately. Any combination of the sorting options may be used. If multiple options are specified, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on.
- n** Sort by filename and extension (default). If **e** is also specified, sort by name only.
 - Reverse the sort order for the next option
 - a** Sort names and extensions in standard ASCII order, rather than sorting numerically when digits are included in the name or extension.
 - c** Sort by compression ratio (the least compressed file in the list will be displayed first).
 - d** Sort by date and time (oldest first); also see [/T:acw](#)
 - e** Sort by extension
 - g** Group subdirectories first, then files
 - i** Sort by file description (ignored if **/C** or **/O:c** is also used).
 - o** Sort by owner
 - r** Reverse the sort order for all options
 - s** Sort by size
 - t** Same as **d**
 - u** Unsorted
- /P** Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /S** Display file information from the current directory and all of its accessible subdirectories.

If you specify a number after the /S, PDIR will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\l\e", /S2 will only affect the "a", "b", and "c" directories.

If you specify a + followed by a number after the /S, PDIR will not display any filenames until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\l\e, /S+2 will not display the contents of \a or \a\b. (Not available in **TCC/LE**.)

/T:type Specifies which single one of the date and time fields below, available on a drive which supports long filenames, should be displayed and used for sorting:

- a** Last access date and time (NTFS volumes).
- c** Creation date and time.
- w** Last write date and time (default).

If /T is not specified, the default is /T:w.

If you append a **u** after the field, DIR will display the file time in UTC.

Note: If more than one time type is specified, the first one specified is used, and all subsequent ones ignored.

/(...) Use this option to define the various fields and display formats you wish to use for each selected entry. The fields may be in any order, and may be repeated. If this option is not used, the output format is identical to that of the [DIR](#) command. If you specify multiple /(...) options, PDIR will insert a space in the output between each one.

4.3.98 PLAYAVI

Purpose: Play Windows .AVI (video clip) files

Format: PLAYAVI [/A /C /S /Vn] *filename*

filename The file to play

/A(synchronous)	/S(yncronous)
/C(enter)	/V(olume)

Usage:

PLAYAVI "plays" an .AVI or Windows video clip file.

Note: This command relies on the capabilities of your Windows configurations, including access to the proper codec. See your Windows documentation for details.

By default, PLAYAVI operates in synchronous mode, which means **TCC** waits for the .AVI file to complete and its window to close before continuing with the next command in a batch file or alias, or prompting you for a new command. You can change this default behavior with the /A option.

Options:

/= Display the PLAYAVI command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in

TCC/LE.)

- /A** Plays the .AVI file in asynchronous mode. Control returns to the **TCC** prompt immediately for a new command or to execute the next command in the current batch file or alias.
- /C** Displays the AVI viewer in the middle of the screen. Without this option, the viewer appears in the upper-left corner of the screen.
- /S** Plays the .AVI file in synchronous mode (this is the default). **TCC** pauses until the file has finished playing and its window closes.
- /V** Sets the volume level. The range is 0 (silent) to 100.

4.3.99 PLAYSOUND

Purpose: Play MP3, .WAV, Midi, and other sound files

Format: PLAYSOUND [/A /M /S /U /Vn] *filename*

filename The file to play

[/A\(synchronous\)](#)

[/U\(n mute\)](#)

[/M\(ute\)](#)

[/V\(volume\)](#)

[/Synchronous](#)

Usage:

PLAYSOUND "plays" MP3, .WAV, Midi and other types of sound files for which Windows has an appropriate codec installed. It determines the file type automatically from its contents, not its file extension, so it can play sound files which have an unknown file extension. If you don't specify any arguments, PLAYSOUND will display its [command dialog](#).

By default, PLAYSOUND operates in synchronous mode, which means **TCC** waits for the sound file to complete and its window to close before continuing with the next command in a batch file or alias, or prompting you for a new command. You can change this default behavior with the [/A](#) switch, described below.

You can cancel the playing of a synchronous sound file by pressing Ctrl-Break while it is playing.

Options:

- /=** Display the PLAYSOUND command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE.**)
- /A** Plays the sound file in asynchronous mode. Control returns to the **TCC** prompt immediately for a new command or to execute the next command in the current batch file or alias.
- /M** Mute the volume.
- /S** Plays the sound file in synchronous mode (this is the default). **TCC** pauses until the file has finished playing and its window closes.

- /U Unmute (restore the previous volume level).
- /V Sets the volume level. The range is 0 (silent) to 100.

4.3.100 PLUGIN

Purpose: Load, unload, or display information about plugins

Format: PLUGIN [/B /C /F /I /K /L /P /U /V] plugin ...

/B (full pathname)	/L(oad)
/C(ommands)	/P(ause)
/F(unctions)	/U(nload)
/I(nfo)	/V(ariables)
/K(eystrokes)	

Usage:

Plugins allow you to write your own internal variables, variable functions, and internal commands, put them in a DLL, and have **TCC** load them at startup. Plugin names will override existing internal names, so you can extend and/or replace internal variables and commands. When **TCC** starts, it will automatically load any plugins in the default directory (the subdirectory PLUGINS\ in the **TCC** installation directory). The plugins will be loaded before the startup file ([TCSTART](#)) are executed.

You can also write keystroke plugins that will be called for every keystroke entered at the command line. A keystroke plugin can perform actions when a specific key is entered, or even change the key before passing it back to the command processor.

If no options are specified, PLUGIN will display the currently loaded plugins and their internal variables, variable functions, and commands.

See the Plugin SDK for more information on developing plugins.

Options:

- /= Display the PLUGIN command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /B Display the full pathnames of the plugins.
- /C Only display internal commands in the plugins.
- /F Only display variable functions in the plugins.
- /I Display information about the specified plugin, including the name, author, author's email and web addresses, description, function list, version and build numbers. The /I option supports wildcards.
- /K Only display keystroke plugins.
- /L Loads the specified plugins. If the filename is *, load all plugins from the default directory (the subdirectory PLUGINS\ in the **TCC** installation directory).

- /P Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /U Unloads the specified plugin. If the filename is *, unloads all plugins.
- /V Only display internal variables in the plugins.

4.3.101 POPD

Purpose: Return to the disk drive and directory at the top of the directory stack

Format: POPD [/X * n]

n The number of directories to pop

[/X \(exclude\)](#)

See also: [DIRS](#), [PUSHD](#), [@DIRSTACK](#) and [Directory Navigation](#).

Usage:

Each time you use the [PUSHD](#) command, it saves the current disk drive and directory on the internal directory stack. POPD restores the most recently saved drive and directory and removes that entry from the stack. You can use these commands together to change directories, perform some work, and return to the starting drive and directory.

Directory changes made with POPD are recorded in the directory history list and can be displayed in the [directory history window](#). Read the section on [Directory Navigation](#) for complete details on this and other directory navigation features.

This example saves and changes the current disk drive and directory with [PUSHD](#), and then restores it. The current directory is shown in the prompt:

```
[c:\] pushd d:\database\test
[d:\database\test] pushd c:\wordp\memos
[c:\wordp\memos] pushd a:\123
[a:\123] popd
[c:\wordp\memos] popd
[d:\database\test] popd
[c:\]
```

You can use the [DIRS](#) command to see the complete list of saved drives and directories (the directory stack).

The POPD command followed by an asterisk [*] clears the directory stack without changing the current drive and directory.

If the directory on the top of the stack is not on the current drive, POPD will switch to the drive and directory on the top of the stack without changing the default directory on the current drive.

Options:

- /X Don't save the current directory to the Directory History list.

4.3.102 POSTMSG

Purpose: Post a message to a window

Format: POSTMSG "title" msg wparam lparam

title	The window title
msg	The message to send
wParam	wParam integer
lParam	lParam integer value

Usage:

POSTMSG allows you to send a Windows message to any window with a caption.

The **title** may contain wildcards, and POSTMSG will send the message to the first window with a matching title.

If **title** begins with a =, it is assumed to be a process ID instead of a title. (Note that this is less reliable than providing a title, as a process can have multiple top-level windows.)

See the Windows SDK documentation for a list of possible messages and their parameters.

4.3.103 PRINT

Purpose: Print the specified file(s) using the application associated with each file's extension

Format: PRINT [/A printer /D printer] filename ...

[/A\(dd\) printer](#) [/D\(elete\) printer](#)

Usage:

Except for plain text files, Windows files cannot be printed without sending them to an associated application for interpretation and formatting. Using the extension for each file you want to print, PRINT determines if a Print action has been defined for that file type. If so, it executes the Print action and sends the file to the application for processing.

For example, if you use the command

```
print myletter.doc
```

PRINT looks up the Print command for .DOC files in the registry and, on most computers, will find that it is associated either with WordPad or Word. It will execute the associated program and send it the file along with the necessary command to print the file and then quit.

If PRINT cannot find a Print command for a file, it displays an error message. If there are additional files in the list you gave it to print, it will go on to the next file in the list.

PRINT accepts piped & redirected input to send to the printer. If there is no *filename*, PRINT will read from STDIN, create a temporary file, and send it to the printer.

PRINT depends on proper [Windows File Associations](#) settings in the registry and proper behavior of the program associated with each file type in order to print the file. If the registry entries or the application

associated with a particular file type are not configured correctly, PRINT may not work as expected.

Options:

- /A Add a connection for the specified printer.
- /D Remove the connection to the specified printer.

4.3.104 PRIORITY

Purpose: Display or set process priority, or suspend or resume a process

Format: PRIORITY [/Q /R /S PID | "title" ABOVE | BELOW | NORMAL | HIGH | IDLE | REALTIME]

ABOVE	Above normal priority
BELOW	Below normal priority
NORMAL	Normal (default) priority
HIGH	High priority
IDLE	Idle priority (only executes when no higher priority task is scheduled)
REALTIME	Realtime priority

[/Q\(quiet\)](#) [/S\(suspend\)](#)
[/R\(resume\)](#)

Usage:

You can specify the process either by the PID or by the window title. If you don't specify either a PID or title, PRIORITY will adjust the priority of the current **TCC** process.

If you do not enter any arguments, PRIORITY displays all of the active processes, their current priority, the module names, and the window titles (if any).

Options:

- /Q Don't display any suspend / resume messages.
- /R Resume the process
- /S Suspend the process

4.3.105 PROCESSMONITOR

Purpose: Monitor process start or end

Format: PROCESSMONITOR [/C [name]]
PROCESSMONITOR name STARTED | ENDED | HUNG n command

name	Full pathname of the process to monitor
n	Number of repetitions (or FOREVER)
command	Command to execute when condition is triggered

[/C\(lear\)](#)

Usage:

The process name can include wildcards. If you do not include a path for **name**, PROCESSMONITOR will only compare the filename part of the process names.

The command line will be parsed and expanded before PROCESSMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. PROCESSMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

HUNG will test the process's main window to see if it is still responding to messages. If the process doesn't respond or call GetMessage within 5 seconds, the condition will be triggered. (This is normally only useful for GUI apps.)

If you don't enter any arguments, PROCESSMONITOR will display the processes it is currently monitoring.

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

PROCESSMONITOR creates three environment variables when a process is STARTED that can be queried by **command**. The variables are deleted after **command** is executed.

_processname The name of the process that was started

_processpid The PID of the process

_processcount The number of times the command has been triggered

For example, if you want to be alerted whenever "myapp" exits:

```
processmonitor myapp ended forever sendmail bob@abc.com Myapp Myapp just shut down!
```

Options:

/C If **name** is specified, remove the monitor for that process name. Otherwise, remove all active process monitors.

4.3.106 PROMPT

Purpose: Change the command line prompt

Format: PROMPT [text]

text Text to be used as the new command line prompt.

See also: [ESET](#) and [SET](#) (the PROMPT command is syntactically equivalent to SET PROMPT).

Usage:

You can change and customize the command line prompt at any time. The prompt can include normal text and system information such as the current drive and directory, the time and date, and the amount of memory available. You can create an informal "Hello, Bob!" prompt or a complex prompt full of impressive information.

The prompt **text** can contain special commands in the form **\$?**, where **?** is one of the characters listed below. Unless otherwise specified, those meta characters are case-independent.

- a** The ampersand character[&].
- b** The vertical bar character [|].
- c** The open parenthesis [().
- d** Current date, in the format: **Fri 01-01-10** (the month, day, and year are formatted according to your current country settings)
- D** Current date, in the format: **Fri Aug 19, 2011**
- e** The ASCII ESC character (decimal 27), necessary for [ANSI](#) commands.
- f** The close parenthesis ()].
- g** The > character.
- h** Backspace over the previous character.
- j** Current date in ISO 8601 format (yyyy-mm-dd).
- l** The < character.
- m** Time in hours and minutes using 24-hour format.
- M** Time in hours and minutes using the default country format.
- n** Current drive letter.
- p** Current drive and directory (lower case).
- P** Current drive and directory (upper case on drives which do not support long filenames; directory names shown in mixed case as stored on the disk on LFN drives).
- q** The = character.
- r** The numeric exit code of the last external command.
- s** The space character.
- t** Current 24-hour time, in the format hh:mm:ss.
- T** Current 12-hour time, in the format hh:mm:ss[a|p].
- u** The current user.
- v** Windows version number, in the format **6.0**.
- w** Current directory, in a shortened format. If the current directory is the root or a first-level subdirectory, it is displayed as-is. If it is second level or deeper, the path is truncated (i.e., "c:\...\config"). (This does not work with UNC names.) \$W and \$w behave like \$P and \$p for displaying upper/lower case.
- xd:** Current directory on drive **d:** in lower case, including the drive letter (uses the actual case of the directory name as stored on the disk for LFN drives.)
- Xd:** Current directory on drive **d:** in upper case, including the drive letter.
- z** Current shell nesting level.
- +** Display one + character for each directory on the [PUSHD](#) directory stack.
- \$** The \$ character.
- CR/LF (go to beginning of a new line).
- ~** (Substitute for **P**). If the environment variable HOME (or HOMEDRIVE + HOME PATH) exists, TCC will compare the variable to the beginning of the current path. If they match, TCC will substitute ~ for the variable part. (If they don't match, ~ is treated like a P.)

For example, to set the prompt to the current date and time, with a ">" at the end:

```
[c:\] prompt $d $t $g  
Thu Aug 18, 2011 10:29:19 >
```

To use the ~ (home) metacharacter:

```
[c:\] set home=c:\users\myself
[c:\] set prompt=[\$~]
[c:\] cd \users\myself\downloads
[~\downloads]
```

The **TCC** prompt can be set in [TCSTART](#) or in any batch file that runs when **TCC** starts.

If you enter PROMPT with no parameters, the prompt will be reset to its default value.

You can include literal text and special characters as well as the value of any [environment](#) variable, [internal variable](#), or [variable function](#) in a prompt. For example, if you want to include the size of the largest free memory block in the command prompt, plus the current drive and directory, you could use this command:

```
[c:\] prompt [(%@dosmem[K]K) $p]
[ (31043K) c:\data]
```

Notice that the @DOSMEM function is shown with two leading percent signs [%]. If you used only one percent sign, the @DOSMEM function would be expanded at once when the PROMPT command was executed, instead of every time the prompt is displayed. As a result, the amount of memory would never change from the value it had when you entered the PROMPT command. You can also use *back quotes* to delay expanding the variable function until the prompt is displayed:

```
prompt ` [(%@dosmem[K]K) $p]`
```

You can use this feature along with the [@EXEC](#) variable function to create a complex prompt which not only displays information but executes commands. For example, to execute an alias which checks battery status each time the prompt is displayed (enter the alias on one line):

```
alias cbatt `if %_apmlife lt 30 beep 440 4 880 4 440 4 880 4`
prompt `%@exec[@cbatt]$p$g`
```

You can include [ANSI](#) escape sequences in the PROMPT by using the built-in ANSI X3.64 support in **TCC**. This example uses ANSI X3.64 sequences to set a prompt that displays the shell level, date, time and path in color on the top line of the screen (enter the command as one line):

```
prompt $e[s$e[1;1f$e[41;1;37m$e[K[$z] $d
Time: $t$h$h$h Path: $p$e[u$e[0;32m$n$g
```

4.3.107 PUSHD

Purpose: Save the current disk drive and directory, optionally changing to a new drive and directory

Format: PUSHD [/X path]

path The name of the new default drive and directory.

[/X \(exclude\)](#)

See also: [DIRS](#), [POPD](#), [@DIRSTACK](#) and [Directory Navigation](#).

Usage:

PUSHD saves the current drive and directory to a "last in, first out" directory stack. The [POPD](#) command returns to the last drive and directory that was saved by PUSHD. You can use these commands together to change directories, perform some work, and return to the starting drive and directory. The [DIRS](#) command displays the contents of the directory stack.

To save the current drive and directory, without changing directories, use the PUSHD command by itself, with no **path**.

If a **path** is specified as part of the PUSHD command, the current drive and directory are saved and PUSHD changes to the specified drive and directory. If the **path** includes a drive letter, PUSHD changes to the specified directory on the new drive without changing the current directory on the original drive.

This example saves the current directory and changes to C:\WORDP\MEMOS, then returns to the original directory:

```
[c:\] pushd \wordp\memos  
[c:\wordp\memos] popd  
[c:\]
```

When you use PUSHD to change to a directory on an LFN drive, you must quote the **path** name if it contains white space or special characters.

PUSHD can also change to a network drive and directory specified with a UNC name (see [File Systems](#) for details).

If PUSHD cannot change to the directory you have specified it will attempt to search the [CDPATH](#) and the [extended directory search](#) database. You can also use [wildcards](#) in the **path** to force an extended directory search. Read the section on [Directory Navigation](#) for complete details on these and other directory navigation features.

Directory changes made with PUSHD are also recorded in the directory history list and can be displayed in the [directory history window](#).

The directory stack can hold up to 2047 characters, or about 100 typical entries (depending on the length of the names). If you exceed this limit, the oldest entry is removed before adding a new entry.

Options:

/X Don't save the current directory to the Directory History list.

4.3.108 QUERYBOX

Purpose: Pops up a dialog box to get an input string from the user and save it in an environment variable

Format: QUERYBOX [/CUE="text" /D /E /Ln /P /POS=top,left /Tn] ["title"] *prompt %%varname*

title Text for the title bar of the dialog box.

prompt Text that will appear inside the dialog box.

varname Variable name where the input will be saved.
/CUE Cue text to display in the input box

/D(igits only)	/P(assword)
/E(dit existing value)	/POS (ition)
/L (maximum Length)	/T(imeout)

See also: [INKEY](#), [INPUT](#), and [MSGBOX](#)

Usage:

QUERYBOX displays a dialog box with a prompt, an optional title, and a string input field. Then it waits for your entry, and places any characters you type into an environment variable. QUERYBOX is normally used in batch files and aliases to get text input.

QUERYBOX is similar to INPUT, except it appears as a popup dialog box. If you prefer to work within the command line window, see the INKEY and INPUT commands.

The /CUE option displays the cue text in light gray in the input box (it disappears as soon as you enter a character).

Standard command line editing keys may be used to edit the input string as it is entered. All characters entered up to, but not including, the carriage return are stored in the variable.

For example, to prompt for a string and store it in the variable NAME:

```
querybox "File Name" Enter a name: %%name
```

If you press **Ctrl-C** or **Ctrl-Break** while QUERYBOX is waiting for input, execution of an alias will be terminated, and execution of a batch file will be suspended while you are asked whether to cancel the batch job. A batch file can handle **Ctrl-C** and **Ctrl-Break** itself with [ON BREAK](#).

QUERYBOX returns a value of zero in the internal variable [%_?](#) after a successful operation, and a non-zero value otherwise (a timeout returns 20, a cancel returns 2). Be sure to save the return value in another variable or test it immediately; because the value of [%_?](#) changes with every internal command.

Options:

- /D** Only accepts numeric values.
- /E** Allows you to edit an existing value. If there is no existing value for **varname**, QUERYBOX allows you to enter a new value.
- /Ln** Sets the maximum number of characters which QUERYBOX will accept to **n**.
- /P** Tells QUERYBOX to echo asterisks, instead of the characters you type.
- /POS** Sets the dialog position. (If you don't specify a position, QUERYBOX will center the dialog in the **TCC** window.)
- /Tn** Wait for a maximum of **n** seconds for a response.

4.3.109 QUIT

Purpose: Terminate the current batch file

Format: QUIT [value]

value The numeric exit code to return to **TCC** or to the previous batch file.

See also: [CANCEL](#) and [EXIT](#).

Usage:

QUIT provides a simple way to exit a batch file before reaching the end of the file. If you QUIT a batch file called from another batch file, you will be returned to the previous file at the line following the original CALL.

This example batch file fragment checks to see if the user entered "quit" and exits if true.

```
input Enter your choice : %%option
if "%option" == "quit" quit
```

QUIT only ends the current batch file. To end all batch file processing, use the [CANCEL](#) command.

If you specify a **value**, QUIT will set the [ERRORLEVEL](#) or exit code to that value. For information on exit codes see the [IF](#) command, and the [%?](#) variable. Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

You can also use QUIT to terminate an alias. If you QUIT an alias while inside a batch file, QUIT will end both the alias and the batch file and return you to the command prompt or to the calling batch file.

4.3.110 RD / RMDIR

Purpose: Remove one or more subdirectories

Format: RD [/I"text" /K /N[et] /Q /R /S] [@file] path...

or

RMDIR [/I"text" /K /N[et] /Q /R /S] [@file] path...

path The name of one or more subdirectories to remove.

@file A text file containing the names of the directories to remove, one per line (see [@file lists](#) for details).

[/I \(match descriptions\)](#)

[/Q\(quiet\)](#)

[/K \(no Recycle Bin\)](#)

[/R\(recycle bin\)](#)

[/N \(disable options\)](#)

[/S\(subdirectories\)](#)

See also: [MD](#).

File Selection

Supports [command dialog](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#). Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Internet: Can be used with [FTP Servers](#).

Usage:

RD and RMDIR are synonyms. You can use either one. If you don't specify any arguments, RD will display its command dialog.

RD removes directories from the directory tree. For example, to remove the subdirectory MEMOS from the subdirectory *WP*:

```
rd \wp\memos
```

Before using RD, you must delete all files and subdirectories (and their files) in the **path** you want to remove. Remember to remove hidden and read-only files as well as normal files (you can use [DEL /Z](#) to delete hidden and read-only files).

You can use wildcards in the **path**.

When removing a directory on an LFN drive, you must quote any **path** which contains white space or special characters.

If RD deletes one or more directories, they will be deleted from the [extended directory search](#) database.

You cannot remove the root directory, the current directory (.), any directory above the current directory in the directory tree, or any directory in use by another process. RD will delete hidden directories, for compatibility with CMD.

You can remove directories on [FTP servers](#). For example:

```
rd ftp://ftp.abc.com/data
```

RD sets two internal variables:

%_rd_dirs	The number of directories deleted
%_rd_errors	The number of errors

(Note that if you do an RD /S, the actual deletions are done by DEL, so check the DEL variables.)

Options:

- /= Display the RD command dialog to help you set the directory and command line options.
You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /I"**text**" Select directories by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the /I immediately, with no intervening spaces. You can select all filenames that have a description with /I"[?]*", or all filenames that do not have a description with /I"[]". Do not use /I with @file lists. See [@file lists](#) for details.
- /K When used with the /S option, this will physically delete files instead of sending them to the Windows Recycle Bin, even if you have the [Delete to Recycle Bin](#) configuration option set.
- /N This option takes two possible arguments:

- e** Don't display errors.
- t** Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*).

- /Q** When used with the **/S** option, this will suppress the prompt before deleting the directories.
- /R** When used with the **/S** option, this will send the deleted files to the Windows Recycle Bin.
- /S** This option is included only for compatibility with CMD, and should be used with **EXTREME CARE!!** It deletes all files (including hidden and system files) in the named directory and all of its subdirectories, then removes all subdirectories. **It can potentially erase all files on a drive with a single command.** You cannot use wildcards with the **/S** option.

Note: Do not use **/S** with @file lists.

4.3.111 REBOOT

Purpose: Reboot the computer, log off Windows, or shut down

Format: REBOOT [/H /K /L /M[0|1] /P /R /S /V /W]

/H(ibernate)	/R(eboot)
/K (lock)	/S(hutdown)
/L(logoff)	/V(erify)
/M(onitor)	/W (standby)
/P(ower off)	

Usage:

REBOOT will log off or shut down the operating system, or completely restart your computer. It normally performs a warm reboot, or a shutdown and restart under Windows.

REBOOT defaults to performing a warm boot, with no prompting. The following example prompts you to verify the reboot, then does a warm boot:

```
reboot /v
```

TCC issues the standard commands to shut down other applications and the Windows before rebooting. Windows may prompt you for additional actions, or even ignore the request altogether depending on which processes are running.

Options:

- /H** Save everything in memory to your hard disk, and shutdown to save power. The desktop is restored to its original state when the computer is restarted.
- /K** Lock the workstation. To unlock, the user must log in.
- /L** Log off Windows, but do not reboot. This option is equivalent to selecting Shutdown from the Start menu, then selecting "Close all programs and log on as a different user" in the shutdown dialog.
- /M** Switch the display to low power (M0) or shut off the display (M1 -- will not work on all

systems). This option will not reboot the computer unless you also include /R.

- /P Log off Windows and turn off the computer.
- /R Reboots the system. This is the default, but is required if you specify /M0 or /M1 and also want to reboot.
- /S Shut down the system, but do not reboot. This is equivalent to selecting Shutdown from the Start menu, then selecting "Shut down the computer" in the shutdown dialog.
- /V Prompt for confirmation (Y or N) before acting.
- /W Save power by turning off the monitor and hard disks. When the computer comes out of standby, the desktop is restored to its original state.

4.3.112 RECYCLE

Purpose: Delete files in the recycle bin or display the recycle bin status

Format: RECYCLE [/D /E /Q /P] [drives ...]

drives Local fixed and removable (non CD-ROM / DVD) drives

<u>/D(elete)</u>	<u>/P(rompt)</u>
<u>/E (no error messages)</u>	<u>/Q(uiet)</u>

Usage:

If you don't specify any drives, RECYCLE will display the recycle bin status, or if /D is specified delete everything in the recycle bin for all local drives.

RECYCLE will empty the recycle bin for an entire drive; there is no way to specify individual files.

Options:

- /D Empty the recycle bin for the specified drive(s).
- /E Suppress all non-fatal error messages, such as "File Not Found." Fatal error messages, such as "Drive not ready," will still be displayed. This option is most useful in batch files.
- /P Prompt the user to confirm each delete operation (at least one drive must be specified).
- /Q Don't display the name of the recycle bin(s). This option is most often used in batch files.

4.3.113 REM

Purpose: Put a comment in a batch file

Format: REM [comment]

comment The text to include in the batch file.

Usage:

The REM command lets you place a remark or comment in a batch file. Batch file comments are useful for documenting the purpose of a batch file and the procedures you have used. For example:

```
rem This batch file provides a
rem menu-based system for accessing
rem word processing utilities.
rem
rem Clear the screen and get selection
cls
```

REM must be followed by a space or tab character, then the comment. **TCC** will ignore everything on the line following the REM, including quotes, redirection symbols, and other commands (see below for the exception to this rule).

If ECHO is ON, the comment is displayed. Otherwise, it is ignored. If ECHO is ON and you don't want to display the line, preface the REM command with an at sign [@].

You can also place a comment in a batch file by starting the comment line with two colons [::]. In essence this creates a batch file "label" without a valid label name.

You can use REM to create a zero-byte file if you use a redirection symbol immediately after the REM command. For example, to create the zero-byte file C:\xyz:

```
rem>xyz
```

(This capability is included for compatibility with CMD. A simpler method for creating a zero-byte file with **TCC** is to use **>filename** as a command, with no actual command before the [>] redirection character.)

4.3.114 REN / RENAME

Purpose: Rename files or subdirectories

Format: REN [/A:[[-][+]rhsadecijopt] /B /E /I"text" /N[enst] /O:[-]adegnrstu /P /Q /S /T] [*@file*]
old_name... *new_name*
or
RENAME [/A:[[-][+]rhsadecijopt] /E /I"text" /N[enst] /O:[-]adegnrstu /P /Q /S /T] [*@file*]
old_name... *new_name*

old_name Original name of the file(s) or subdirectory.

new_name New name to use, or new path on the same drive.

@file A text file containing the names of the source files to rename, one per line
(see [@file lists](#) for details).

[/A: \(Attribute select\)](#)

[/O:... \(Order\)](#)

[/B \(Rename on reboot\)](#)

[/P\(rompt\)](#)

[/E \(No error messages\)](#)

[/Q\(uiet\)](#)

[/I"text" \(match description\)](#)

[/S\(ubdirectory\)](#)

[/MD \(Create target directory\)](#)

[/T\(otal\)](#)

[/N \(Disable\)](#)

See also: [COPY](#) and [MOVE](#).

File Selection:

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), [delayed variable expansion](#), and [include lists](#). Use wildcards with caution on LFN volumes; see [LFN File Searches](#) for details.

Internet: Can be used with [FTP/HTTP Servers](#) and HTTP/HTTPS servers.

Usage:

REN and RENAME are synonyms. You may use either one. If you don't specify any arguments, REN will display its command dialog.

REN lets you change the name of a file or a subdirectory, or move one or more files to a new subdirectory on the same drive. New files may be on different file systems or drives; new directories must be on the same drive.

In its simplest form, you give REN the ***old_name*** of an existing file or subdirectory and then a ***new_name***. The ***new_name*** must not already exist; you can't give two files the same name (unless they are in different directories). The first example renames the file *MEMO.TXT* to *MEM.TXT*. The second example changes the name of the *\WORD* directory to *\WP*:

```
rename memo.txt mem.txt
rename /s \word \wp
```

When you rename files or directories on an LFN drive, you must quote any names which contain white space or special characters.

You can also use REN to rename a group of files that you specify with wildcards, as multiple files, or in an include list. When you do, the ***new_name*** must use one or more wildcards to show what part of each filename to change. Both of the next two examples change the extensions of multiple files to .SAV:

```
ren config.nt autoexec.nt tcstart.btm *.sav
ren *.txt *.sav
```

REN can move files to a different subdirectory on the same drive. When it is used for this purpose, REN requires one or more filenames for the ***old_name*** and a directory name for the ***new_name***:

```
ren memo.txt \wp\memos\
ren oct.dat nov.dat \data\save\
```

The final backslash in the last two examples is optional. If you use it, you force REN to recognize the last parameter as the name of a directory, not a file. The advantage of this approach is that if you accidentally mistype the directory name, REN will report an error instead of renaming your files in a way that you didn't intend.

REN can also move files to a new directory and change their name at the same time if you specify both a path and file name for ***new_name***. In this example, the files are renamed with an extension of .SAV as they are moved to a new directory:

```
ren *.dat \data\save\*.sav
```

If you use REN to rename a directory, the ***new_name*** must normally be specified explicitly, and cannot contain wildcards. You can override this restriction with **/S**. When you rename a directory the [extended](#)

[directory search](#) database will be automatically updated to reflect the change.

You can also rename a subdirectory to a new location in the directory tree on the same physical drive (sometimes called "prune and graft"). You must specify the new name explicitly, not just give the path. For example, if the **D:\TCMD** directory contains a subdirectory *TEST*, you can rename *TEST* to be a subdirectory of the root directory like this:

```
[d:\tcmd] ren TEST \TEST\
```

REN does not change a file's attributes, except to set attribute **A**. The **new_name** file(s) will have the same attributes as **old_name**.

If you have appropriate permissions, you can rename files on FTP, HTTP, and HTTPS servers. For example:

```
ren ftp://ftp.abc.com/file1.txt file2.txt
```

Wildcard characters like [*] and [?] will be treated as wildcards in FTP URLs, but will be treated as normal characters in HTTP URLs.

You can also use the IFTP command to start an FTP session on a server, and then use an abbreviated syntax to specify the files and directories you want. For more information, see [Using FTP/HTTP Servers and IFTP](#).

REN supports [regular expression](#) back references in the target name. If you are using back references, you must also use a regular expression in the source name. The syntax is:

```
ren ::filename ::target
```

REN sets three internal variables:

%_ren_dirs	The number of directories renamed
%_ren_files	The number of files renamed
%_ren_errors	The number of errors

Note: The wildcard expansion process will attempt to allow both CMD-style "extension" matching (assumes only one extension, at the end of the word) and the advanced **TCC** string matching (allowing things like *.*.abc) when an asterisk is encountered in the destination of a REN command.

Options:

- /= Display the REN command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A: Rename only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with [@file lists](#). See [@file lists](#) for details.

You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /B If REN can't rename the file (i.e., access denied), it will schedule it to be renamed at the next reboot.

/E Suppress all non-fatal error messages, such as "File Not Found." Fatal error messages, such as "Drive not ready," will still be displayed. This option is most useful in batch files.

/I"*text*" Select files by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the **/I** immediately, with no intervening spaces. You can select all filenames that have a description with **/I"[?]"**, or all filenames that do not have a description with **/I"[]"**. Do not use **/I** with @file lists. See [@file lists](#) for details.

/MD Create the target directory if it doesn't exist. Note that you *must* either terminate the target directory name with a trailing \ or specify a filename component; otherwise REN cannot tell what you want for the directory and what you want for the filename.

/N Do everything except actually rename the file(s). **/N** displays how many files would be renamed. This option is useful for testing what a REN command will actually do.

A **/N** with one or more of the following arguments has an alternate meaning:

- e** Don't display errors.
- n** Don't update the file descriptions
- s** Don't display the summary
- t** Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*).

/O:... Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

The **/O:...** option saves all of the matching filenames and then performs the rename. This avoids the potential problem of renaming files more than once.

/P Prompt the user to confirm each rename operation. Your options at the prompt are explained in detail under [Page and File Prompts](#).

/Q Don't display filenames or the number of files renamed. When used in combination with the **/P** option above, it will prompt for filenames but will not display the totals. This option is most often used in batch files. See also **/T**.

- /S** Normally, you can rename a subdirectory only if you do not use any wildcards in the **new_name**. This prevents subdirectories from being renamed inadvertently when a group of files is being renamed with wildcards. **/S** will let you rename a subdirectory even when you use wildcards. **/S** does not cause REN to process files in the current directory and all subdirectories as it does in some other file processing commands. To rename files throughout a directory tree, use [GLOBAL](#) REN.
- /T** Don't display filenames as they are renamed, but report the number of files renamed. See also [/Q](#).

4.3.115 RESOLUTION

Purpose: Change the resolution of the specified display

Format: **RESOLUTION [displayname] width height [depth [frequency]]**

displayname The name of the monitor to update
width The new display width in pixels
height The new height in pixels
depth The new color depth
frequency The new refresh frequency

Usage:

If you don't specify any arguments, RESOLUTION will display the display devices and monitors.

4.3.116 RETURN

Purpose: Return from a GOSUB (subroutine) in a batch file

Format: **RETURN [value]**

value The numeric exit code to return to **TCC**

See also: [GOSUB](#).

Usage:

TCC allows subroutines in batch files.

A subroutine begins with a label (a colon followed by one or more words) and ends with a RETURN command.

The subroutine is invoked with a GOSUB command from another part of the batch file. When a RETURN command is encountered the subroutine terminates, and execution of the batch file continues on the line following the original GOSUB. If RETURN is encountered without a GOSUB, **TCC** will display a "Missing GOSUB" error message.

You cannot execute a RETURN from inside a [DO](#) loop.

The following batch file fragment calls a subroutine which displays the files in the current directory:

```
echo Calling a subroutine
gosub subr1
echo Returned from the subroutine
quit
:subr1
dir /a/w
return
```

If you specify a **value**, RETURN will set the internal exit code to that value. That exit code should be tested immediately upon return from the subroutine and before it is reset by another command. For information on exit codes from internal commands, see the [?](#) variable.

4.3.117 REXEC

Purpose: Remotely execute commands

Format: REXEC [/H host /U name /P password /Rn /Tn] host [/L userid] command ...

command The command to execute

/H(host name)	/R(emote port)
/L(user ID)	/T(firewall type)
/P(password)	/U(username)

Usage:

RExec allows remote execution of commands on any system with the rexec service installed. Press Ctrl-C to disconnect from the other system.

If you don't specify a username, REXEC will use the current username. You can provide a password on the command line by appending it to the username (i.e., "User:Password"). If you don't provide a password, REXEC will prompt for it.

If you want to do redirection on the remote system, enclose the argument list in double quotes. For example:

```
RExec /H host /U user /P password "command | command2"
```

The double quotes will be removed before passing the commands to the remote system.

Note: Windows does not include the rexec service, so you will need to get one from a third-party and install it on the remote system before executing REXEC.

Options:

/H	Firewall host name
/L	User name (ID)
/P	Firewall user password
/R	Remote port number
/T	Firewall type, where <i>n</i> is:

- 0 No firewall (default setting)
- 1 Connect through a tunneling proxy
- 2 Connect through a SOCKS4 Proxy
- 3 Connect through a SOCKS5 Proxy

/U Firewall user name

4.3.118 RSHELL

Purpose: Remotely execute commands

Format: RSHELL [/H host /U name /P password /Rn /Tn] host [/L userid] command ...

command The command to execute

[/H\(host name\)](#)

[/R\(emote port\)](#)

[/L \(user ID\)](#)

[/T \(firewall type\)](#)

[/P\(password\)](#)

[/U\(username\)](#)

Usage:

RSHELL allows remote execution of commands on any system with the rshell service installed. Press Ctrl-C to disconnect from the other system.

If you don't specify a username, RSHELL will use the current username.

If you want to do redirection on the remote system, enclose the argument list in double quotes. For example:

RSHELL /H host /U user /P password "command | command2"

The double quotes will be removed before passing the commands to the remote system.

Note: Windows does not include the rshell service, so you will need to get one from a third-party and install it on the remote system before executing RSHELL.

Options:

/H Firewall host name

/L User name

/P Firewall user password

/R Remote port number

/T Firewall type, where *n* is:

- 0 No firewall (default setting)
- 1 Connect through a tunneling proxy
- 2 Connect through a SOCKS4 Proxy
- 3 Connect through a SOCKS5 Proxy

/U Firewall user name

4.3.119 SCREEN

Purpose: Position the cursor on the screen and optionally display a message

Format: SCREEN row column [text]

row The new row location for the cursor
column The new column location for the cursor
text Optional text to display at the new cursor location

See also: [ECHO](#) and [ECHOERR](#), [ECHOS](#) and [ECHOSEERR](#), [SCRPUT](#), [TEXT](#), and [VSCRPUT](#).

Usage:

SCREEN allows you to create attractive screen displays in batch files. SCRPUT allows you to specify where a message will appear on the screen. You can use SCREEN to create menus and other similar displays. For example, the following batch file fragment displays a menu:

```
@echo off
cls
screen 3 10 Select a number from 1 to 4:
screen 6 20 1 - Word Processing
screen 7 20 2 - Spreadsheet
screen 8 20 3 - Telecommunications
screen 9 20 4 - Quit
```

SCREEN does not change the screen colors. To display text in specific colors, use [SCRPUT](#) or [VSCRPUT](#). SCREEN always leaves the cursor at the end of the displayed text.

The **row** and **column** values are zero-based, so on a 25 line by 80 column display, valid **rows** are 0 - 24 and valid **columns** are 0 - 79. SCREEN checks for a valid **row** and **column**, and displays a "Usage" error message if either value is out of range.

In **TCC**, the maximum **row** value is determined by the current height of the **TCC** tab window, and the maximum **column** value is determined by the current virtual screen width (see [Resizing the Take Command Window](#) for more information).

You can also specify the **row** and **column** as offsets from the current cursor position. Begin the value with a plus sign [+] to move the cursor down or to the right, or with a minus sign [-] to move the cursor up or to the left. This example prints a string 3 lines above the current position, in absolute column 10:

```
screen -3 10 Hello, World!
```

you specify 999 for the **row**, SCREEN will center the text vertically on the display. If you specify 999 for the **column**, SCREEN will center the text horizontally. This example prints a message at the center of the **TCC** tab window:

```
screen 999 999 Hello, World
```

4.3.120 SCREENMONITOR

Purpose: Monitor the Windows screen saver

Format: SCREENMONITOR [/C]
SCREENMONITOR n command

n Number of repetitions (or **FOREVER**)
command Command to execute when the Windows screen saver is activated

[/C\(lear\)](#)

Usage:

SCREENMONITOR will set its trigger when the Windows screen saver is activated.

If you don't enter any arguments, if SCREENMONITOR is active it will display the repeat count and the command.

The command line will be parsed and expanded before SCREENMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. SCREENMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

Options:

/C Remove the screen saver monitor.

4.3.121 SCRIPT

Purpose: Run a script using an Active Scripting engine

Format: SCRIPT [/E engine] [filename ...]

[/E\(ngine\)](#)

engine The name of the scripting engine

Usage:

If you don't specify any arguments, SCRIPT will display the installed engines.

See also the [@SCRIPT](#) variable function.

TCC has a COM interface to allow the script to call back into **TCC**. The methods are:

shell.exec("command") - execute the specified command (internal or external)

shell.write("string") - write the string to stdout

shell.writeln("string") - write the string to stdout with a CR/LF

shell.alert("text") - pop up a message box

Options:

- /E** If the script doesn't have a recognized extension (i.e., **.vbs**, **.pls**, etc.) you will need to specify the engine SCRIPT should use to execute the script.

4.3.122 SCRPUT

Purpose: Position text on the screen and display it in color

Format: SCRPUT *row col [BRight] fg ON [BRight] bg text*

row	Starting row
col	Starting column
fg	Foreground character color
bg	Background character color
text	The text to display

See also: [ECHO and ECHOERR](#), [ECHOS and ECHOERR](#), [SCREEN](#), [TEXT](#), and [VSCRPUT](#).

Usage:

SCRPUT allows you to create attractive screen displays in batch files. SCRPUT allows you to specify where a message will appear on the screen and what colors will be used to display the message text. You can use SCRPUT to create menu displays, logos, etc.

SCRPUT works like SCREEN, but requires you to specify the display colors. See [Colors and Color Names](#) for details.

The **row** and **column** values are zero-based, so on a 25 line by 80 column display, valid **rows** are 0 - 24 and valid **columns** are 0 - 79. The maximum **row** value is determined by the current height of the **TCC** tab window. The maximum **column** value is determined by the current virtual screen width (see [Resizing the Take Command Window](#) for more information).

SCRPUT checks for a valid **row** and **column**, and displays a "Usage" error message if either value is out of range.

You can also specify the **row** and **column** as offsets from the current cursor position. Begin the value with a plus sign [**+**] to move down the specified number of rows or to the right the specified number of columns, or with a minus sign [**-**] to move up or to the left.

If you specify 999 for the **row**, SCRPUT will center the text vertically in the **TCC** tab window. If you specify 999 for the **column**, SCRPUT will center the text horizontally.

SCRPUT does not move the cursor when it displays the **text**.

The following batch file fragment displays part of a menu, in color:

```
cls white on blue
scrput 3 10 bri whi on blu Select an option:
scrput 6 20 bri red on blu 1 - Word Processing
scrput 7 20 bri yel on blu 2 - Spreadsheet
scrput 8 20 bri gre on blu 3 - Communications
scrput 9 20 bri mag on blu 4 - Quit
```

4.3.123 SELECT

Purpose: Interactively select files for a command

Format: `SELECT [/1 /A[:][-][+]rhsadecijopt] /C /D /E /H /"text" /J /L /O:[-]acdeginorsu /Q /T:acw /X /Z] [command] ... (files....)...`

command The command to execute with the selected files.

files The files from which to select. File names may be enclosed in either parentheses or square brackets. The difference is explained below.

/1 One selection only	/J(ustify names)
/A(tribute select)	/L(ower case)
/C(ompression)	/O(rder)
/D(isable color coding)	/Q (owner)
/E (use upper case)	/T(ime)
/H(ide dots)	/X(display short names)
/I"text" (match descriptions)	/Z (FAT format)

File Selection

Supports [command dialog](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#). Ranges **must** appear immediately after the SELECT keyword.

Internet: Can be used with FTP servers. See [Using FTP/HTTP Servers](#).

Usage:

SELECT allows you to select files for internal and external commands by using a "point and shoot" display. You can have SELECT execute a command once for each file you select, or have it create a list of files for a command to work with. The **command** can be an internal command, an alias, an external command, or a batch file. If you don't specify any arguments, SELECT will display its command dialog.

If you use parentheses around the **files**, SELECT executes the **command** once for each file you have selected. During each execution, one of the selected files is passed to the **command** as a parameter. If you use square brackets around **files**, the SELECTed files are combined into a single list, separated by spaces. The command is then executed once with the entire list presented as part of its command line parameters.

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. SELECT will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

SELECT can also select files on FTP servers. For example:

```
select del (ftp://ftp.domain.com/)
```

You can also use the IFTP command to start an FTP session on a server, and then use an abbreviated syntax to specify the files and directories you want. For more information, see [Using FTP/HTTP Servers](#) and [IFTP](#).

Using the SELECT File List

When you execute the SELECT command, the file list is displayed in a full-window format which includes a top-line status bar and shows the command to be executed, the number of files marked, and the number of Kbytes in those files.

SELECT supports the mouse for selecting and scrolling the list. You can also use the cursor up, cursor down, PgUp, and PgDn keys to scroll through the file list. You can also use character matching to find specific files, just as you can in any [popup window](#). While the file list is displayed you can enter any of the following keys to select or unselect files, display files, execute the command, or exit:

space	Select a file, or unselect a marked file
+	Select a file (all products), or unselect a marked file
-	Unselect a marked file
*	Reverse all of the current marks (except those on subdirectories). If no files have been marked you can use * to mark all of the files
/	Unselect all files
Ctrl-L	View the current highlighted file with LIST . When you exit from LIST, the SELECT screen will be restored
Enter	Execute the command with the marked files, or with the currently highlighted file if no files have been marked
Esc	Skip the files in the current display and go on to the next file specification inside the parentheses or brackets (if any)
Ctrl-C or Ctrl-Break	Cancel the current SELECT command entirely

On FAT drives the file list is shown in standard FAT directory format, with names at the left and descriptions at the right. On LFN drives the format is similar but more space is allowed for the name, and the description is not shown. In this format long names are truncated if they do not fit in the allowable space. For a short-name format (including descriptions) on long filename drives, use the **/X** and **/Z** switches.

When displaying descriptions in the short filename format, SELECT adds a right arrow at the end of the line if the description is too long to fit on the screen. This symbol will alert you to the existence of additional description text. You can use the left and right arrow keys to scroll the description area of the screen horizontally and view the additional text.

Creating SELECT Commands

In the simplest form of SELECT, you merely specify the command and then the list of files from which you will make your selection(s). For example:

```
select copy (*.cmd *.exe) q:\
```

will let you select from among the .CMD files in the current directory, and will then invoke the COPY

command to copy each file you select to the root of drive Q:. After the *.CMD* files are done, the operations will be repeated for the *.EXE* files.

If you want to select from a list of all the *.CMD* and *.EXE* files mixed together, create an [include list](#) inside the parentheses by inserting a semicolon:

```
select copy (*.cmd;*.exe) a:\
```

Finally, if you want the SELECT command to send a single list of files to COPY, instead of invoking COPY once for each file you select, put the file names in square brackets instead of parentheses:

```
select copy [*.*] a:\
```

If you use brackets, you have to be sure that the resulting command (the word COPY, the list of files, and the destination drive in this example) does not exceed the [command line length limit](#). The current line length is displayed by SELECT while you are marking files to help you to stay within that limit.

The parentheses or brackets enclosing the file name(s) can appear anywhere within the command; SELECT assumes that the first set of parentheses or brackets it finds is the one containing the list of files from which you wish to make your selection.

When you use SELECT on an LFN drive, you must quote any file names inside the parentheses which contain white space or special characters. For example, to copy selected files from the **Program Files**" directory to the E:**!SAVE** directory:

```
select copy ("Program Files\*") e:\save\
```

File names passed to the **command** will be quoted automatically if they contain white space or special characters.

The list of files from which you wish to select can be further refined by using [date, time, size and file exclusion ranges](#). The range(s) must be placed immediately after the word SELECT. If the **command** is an internal command that supports ranges, an independent range can also be used in the **command** itself.

You cannot use command grouping to make SELECT execute several commands, because SELECT will assume that the parentheses are marking the list of files from which to select, and will display an error message or give incorrect results if you try to use parentheses for command grouping instead. (You can use a SELECT command inside command grouping parentheses, you just can't use command grouping to specify a group of commands for SELECT to execute.)

Advanced Topics

If you don't specify a command, the selected filename(s) will become the command. For example, this command defines an alias called UTILS that selects from the executable files in the directory C:**UTIL**, and then executes them in the order marked:

```
alias utils select (c:\util\*.cmd;*.exe;*.btm;*.bat)
```

If you want to use [filename completion](#) to enter the filenames inside the parentheses, type a space after the opening parenthesis. Otherwise the command line editor will treat the open parenthesis as the first character of the filename.

With the **/I** option, you can select files based on their descriptions. SELECT will display files if their description matches the text after the **/I** switch. The search is not case sensitive. You can use wildcards and extended wild cards as part of the text.

When sorting file names and extensions for the SELECT display, **TCC** normally assumes that sequences of digits should be sorted numerically (for example, the file DRAW2 would come before DRAW03 because 2 is numerically smaller than 03), rather than strictly alphabetically (where DRAW2 would come second because "2" comes after "0"). You can defeat this behavior and force a strict alphabetic sort with the **/O:a** option.

Options:

- /1** Only allow one selection.
- /A[:]** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow **/A:**.
You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /C** Display per-file and total compression ratios on compressed drives. The compression ratio is displayed instead of the file description. The ratio is left blank for directories and files with a length of 0 bytes, and for files on non-compressed drives. The compression ratios will not be visible on LFN drives unless you use **/Z** to switch to the short filename format. Only compressed NTFS drives are supported. See [DIR /C](#) for more details on how compression ratios are calculated.
- /D** Temporarily turn off directory colorization.
- /E** Display filenames in upper case.
- /H** Suppress the display of the **".** and **".."** directory names.
- /I"text"** Display filenames by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the **/I** immediately, with no intervening spaces. You can select all filenames that have a description with **/I"[?]*"**, or all filenames that do not have a description with **/I"["]**.
- /J** Justify (align) filename extensions and display them in the FAT format.
- /L** Display file and directory names in lower case.
- /O** Set the sort order for the files. The order can be any combination of the following options:
 - n** Sort by filename (this is the default)
 - Reverse the sort order for the next option.
 - a** Sort names and extensions in standard ASCII order, rather than sorting numerically when digits are included in the name or extension.
 - c** Sort by compression ratio (the least compressed file in the list will be displayed first). For information on supported compression systems see **/C** above.
 - d** Sort by date and time (oldest first).
 - e** Sort by extension.
 - g** Group subdirectories together.

- i** Sort by the file description (ignored if **/C** or **/O:c** is also used).
 - o** Sort by owner
 - r** Reverse the sort order for all options.
 - s** Sort by size.
 - u** Unsorted.
- /Q** Display the file owner (requires > 80 column display).
- /T:acw** Specify which of the date and time fields on an LFN drive should be displayed and used for sorting:
- a** Last access date and time (access time is not saved on VFAT and FAT32 volumes).
 - c** Creation date and time.
 - w** Last write date and time (default).
- /X** Display short filenames in FAT format (like **/Z**), on LFN drives.
- /Z** Display a directory on an LFN drive in the old-style format, with the filename at the left and the description at the right. Long names will be truncated to 12 characters; if the name is longer than 12 characters, it will be followed by a right arrow.

4.3.124 SENDHTML

Purpose: Send an HTML-formatted email message

Format: SENDHTML [/A file1 [/A file2 ...] /D /Eaddress /H"header: value" /In /M /Pn /R /
SMTP=server /Sn /SSL=[n] /USER=address /V /X] "address[,address...]"
[cc:address[,address...]]" subject [text | @msgfile]

file1...	The attachment files
address	The destination email address
subject	The subject line
text	The message to send
msgfile	The file containing the message body
/SSL=n	SSL negotiation type
/SMTP=server	Override the default SMTP server
/USER=address	Override the default sending email account

/A file	Attachment	/Pn	Priority
/D	Delivery Confirmation	/R	Send read receipt
/E	Reply-to address	/Sn	Sensitivity
/H	Send custom header	/V	Verbose
/In	Importance	/X	EHLO
/M	CRAM-MD5 authentication		

See also: [SNPP](#) and [SMPP](#).

Usage:

SENDHTML sends an HTML email message from **TCC** via SMTP. The text of the message can be entered either on the command line or read from a text file. SENDHTML also supports SMTP over SSL. If you don't specify any arguments, SENDHTML will display its command dialog.

Before you can use SENDHTML, you must either set the [SMTP](#) configuration options, or have a default account in the registry. Depending on your system configuration, you may also need to start an Internet connection before you use SENDHTML.

A SENDHTML message has three required parts: an [address](#), a [subject](#), and [message](#). Optionally it may also have [attachments](#).

1. The **address** field contains one or more standard Internet email addresses:

```
sendhtml abc@xyz.com ...
```

If **address** contains white space, the entire address field must be surrounded by quotes. You can specify multiple destinations by separating the addresses with **commas** and enclosing the entire string in quotes (all addresses will appear in the "To:" header sent to all recipients). You can add **cc** (**copy**) addresses by prefacing the desired target(s) with **cc:**; and **BCC** (**blind copy**) addresses by prefacing the desired target(s) with **bcc:**. For example:

```
sendhtml "bob@bob.com bcc:joe@joe.com" Test Hello!
```

will send the text **Hello!** with subject **Test** to **bob@bob.com** with a blind copy to **joe@joe.com**.

2. The **subject** will appear as the subject line in the message. If it contains white space, it must be surrounded by quotes.

3. The **message** may either be entered on the command line, or it may be placed in a text file. To tell SENDHTML to send the contents of a file as the message text, use @ sign, followed by the filename. You can use the same approach to send the text content of the clipboard (@CLIP:) or the console (@CON:):

```
sendhtml abc@xyz.com Party @c:\messages\invitation.txt  
sendhtml abc@xyz.com Party @clip:  
type myfile.txt | sendmail abc@xyz.com Party @con:
```

Options:

/= Display the SENDHTML command dialog to help you set the command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

/A file Attach **file** to the email message. The **/A** switch and the name of the file to attach must appear *before address*. Any file name that contains spaces or special characters must be quoted. You can send multiple files by repeating the **/A** switch for each additional file to send. For example:

```
sendhtml /a file1 /a "d:\path\My file2" abc@xyz.com ...
```

/D Request Delivery Notification.

/E Set the "reply to" address in the message header.

/H Set a custom header. The header will be appended to the message headers created from "to", "from", "subject", etc. The headers must of the format "header: value" as specified in RFC 822. You can specify multiple headers with multiple **/H** arguments.

/In Set the Importance where *n* is:

- 1** High
- 2** Normal (default)
- 3** Low

/M Use CRAM-MD5 authentication.

/Pn Set the Priority where *n* is:

- 0** Unspecified (default)
- 1** Normal
- 2** Urgent
- 3** Non Urgent

/R (Read receipt) : Send a read receipt.

/Sn Set the message sensitivity. The values are:

- 1** Personal
- 2** Private
- 3** CompanyConfidential

/SMTP Overrides the default SMTP server (as set in the registry) to use when sending mail.

/SSL=n Type of SSL negotiation. The values are:

- 0** Automatic (default if no *n* value is specified). If the remote port is set to the standard plaintext port, SENDHTML will use Explicit mode. In all other cases, SSL negotiation will be implicit.
- 1** Implicit - SSL negotiation will start immediately after the connection is established.
- 2** Explicit - SENDMAIL will first connect in plaintext, and then explicitly start SSL negotiation.
- 3** No SSL negotiation or security. (This is the default if /SSL is not specified.)

/USER Overrides the default email account (as set in the registry) to use when sending mail.

/V Show all the interaction with the server, except the message header and message body text.

/X Send EHLO instead of HELO.

4.3.125 SENDMAIL

Purpose: Send an email message

Format: SENDMAIL [/A file1 [/A file2 ...] /D /Eaddress /H"header: value" /In /M /Pn /R /Sn /
SMTP=server /SSL[=n] /USER=address /V /X] "address[,address...] [cc:address[,address]
bcc:address[,address...]]" subject [text | @msgfile]

<u>file1...</u>	The attachment files
<u>address</u>	The destination email address
<u>subject</u>	The subject line
<u>text</u>	The message to send
<u>msgfile</u>	The file containing the message body
<u>/SSL=n</u>	SSL negotiation type
<u>/SMTP=server</u>	Override the default SMTP server
<u>/USER=address</u>	Override the default sending email account
<u>/A</u> file	Attachment
<u>/D</u>	Delivery Confirmation
<u>/E</u>	Reply-to address
<u>/H</u>	Send custom header
<u>/In</u>	Importance
<u>/M</u>	CRAM-MD5 authentication
<u>/Pn</u>	Priority
<u>/R</u>	Send read receipt
<u>/Sn</u>	Sensitivity
<u>/V</u>	Verbose
<u>/X</u>	Send EHLO

See also: [SNPP](#) and [SMPP](#).

Usage:

SENDMAIL sends an email message from **TCC** via SMTP. The text of the message can be entered either on the command line or read from a text file. SENDMAIL also supports SMTP over SSL. If you don't specify any arguments, SENDMAIL will display its command dialog.

Before you can use SENDMAIL, you must either set the [SMTP](#) configuration options, or have a default account in the registry. Depending on your system configuration, you may also need to start an Internet connection before you use SENDMAIL.

A SENDMAIL message has three required parts: an [address](#), a [subject](#), and [message](#). Optionally it may also have [attachments](#).

1. The **address** field contains one or more standard Internet email addresses:

```
sendmail abc@xyz.com ...
```

If **address** contains white space, the entire address field must be surrounded by quotes. You can specify multiple destinations by separating the addresses with **commas** and enclosing the entire string in quotes (all addresses will appear in the "To:" header sent to all recipients). You can add **cc** (**copy**) addresses by prefacing the desired target(s) with **cc:;** and **BCC** (**blind copy**) addresses by prefacing the desired target(s) with **bcc:.** For example:

```
sendmail "bob@bob.com bcc:joe@joe.com" Test Hello!
```

will send the text **Hello!** with subject **Test** to **bob@bob.com** with a blind copy to **joe@joe.com**.

2. The **subject** will appear as the subject line in the message. If it contains white space, it must be surrounded by quotes.

3. The **message** may either be entered on the command line, or it may be placed in a text file. To tell SENDMAIL to send the contents of a file as the message text, use @ sign, followed by the filename. You can use the same approach to send the text content of the clipboard (@CLIP:) or the console (@CON:):

```
sendmail abc@xyz.com Party @c:\messages\invitation.txt
sendmail abc@xyz.com Party @clip:
type myfile.txt | sendmail abc@xyz.com Party @con:
```

Options:

/= Display the SENDMAIL command dialog to help you set the command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

/A file Attach **file** to the email message. The **/A** switch and the name of the file to attach must appear *before address*. Any file name that contains spaces or special characters must be quoted. You can send multiple files by repeating the **/A** switch for each additional file to send. For example:

```
sendmail /a file1 /a "d:\path\My file2" abc@xyz.com ...
```

/D Request Delivery Notification.

/E Set the "reply to" address in the message header.

/H Set a custom header. The header will be appended to the message headers created from "to", "from", "subject", etc. The headers must of the format "header: value" as specified in RFC 822. You can specify multiple headers with multiple **/H** arguments. For example, to send HTML mail:

```
sendmail /h"Content-Type: text/html" ...
```

/In Set the Importance where **n** is:

- 1** High
- 2** Normal (default)
- 3** Low

/M Use CRAM-MD5 authentication.

/Pn Set the Priority where **n** is:

- 0** Unspecified (default)
- 1** Normal
- 2** Urgent
- 3** Non Urgent

/R (Read receipt) : Send a read receipt.

/Sn Set the message sensitivity. The values are:

- 1** Personal
- 2** Private
- 3** CompanyConfidential

/SMTP Overrides the default SMTP server (as set in the registry) to use when sending mail.

/SSL=n Type of SSL negotiation. The values are:

- 0** Automatic (default if no *n* value is specified). If the remote port is set to the standard plaintext port, SENDMAIL will use Explicit mode. In all other cases, SSL negotiation will be implicit.
- 1** Implicit - SSL negotiation will start immediately after the connection is established.
- 2** Explicit - SENDMAIL will first connect in plaintext, and then explicitly start SSL negotiation.
- 3** No SSL negotiation or security. (This is the default if /SSL is not specified.)

- /USER** Overrides the default email account (as set in the registry) to use when sending mail.
- /V** Show all the interaction with the server, except the message header and message body text.
- /X** Send EHLO instead of HELO.

4.3.126 SERVICEMONITOR

Purpose: Monitor service start, pause, and / or stop

Format: SERVICEMONITOR [/C [name]]
SERVICEMONITOR name STARTED | PAUSED | STOPPED n command

name Device name
n Number of repetitions (or **FOREVER**)
command Command to execute when condition is triggered

[/C\(lear\)](#)

Usage:

The service name can include wildcards.

The command line will be parsed and expanded before SERVICEMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. SERVICEMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing).

If you don't enter any arguments, SERVICEMONITOR will display the services it is currently monitoring.

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

SERVICEMONITOR creates several environment variables when a service is started, paused, or stopped that can be queried by **command**. The variables are deleted after **command** is executed.

_servicedisplay Display name used by service control programs to identify the service

_servicename The name of the service in the service control manager database

_servicecount The number of times the command has been triggered

_servicestate The current state of the service. The possible values are:

- 1 The service is stopped
- 2 The service is starting
- 3 The service is stopping
- 4 The service is running
- 5 The service continue is pending
- 6 The service pause is pending
- 7 The service is paused

Options:

- /C** If **name** is specified, remove the monitor for that service. Otherwise, remove all service monitors.

4.3.127 SERVICES

Purpose: Display, stop, or start system services

Format: SERVICES [/P /R /S] [name ...]

/P(ause) /S(top service)
/R(un)

Usage:

The **name** is the service name, not the display name. **name** can contain wildcards.

You must be an admin user to run or stop a service.

Options:

/P Pause after displaying each page.

/R Run the specified service(s).

/S Stop the specified service(s).

4.3.128 SET

Purpose: Display, create, modify, or delete environment variables

Format: Display mode:

SET [/D /E /P /S /U /V /X] [wildname]

Definition mode:

SET [/A /D /O /S /U /V /E /RO /R [file...]] | name=value | prompt]

Deletion mode:

SET [/D /S /U /V /E] name=

file	One or more input files from which to read variable definitions.
name	The name of the environment variable.
value	The new value for the variable, separated from name by space[s].
prompt	Optional input prompt for the /P name= option.
wildname	Name of variable[s] to be displayed. May contain * wildcard unless displaying registry variables.

/A	Arithmetic	/R	Read from file(s)
/D	Default	/RO	Readonly variable
/E	Environment, too	/S	System
/P	Pause or Prompt	/U	User
/O	Don't overwrite	/V	Volatile
/Q	Don't echo /A result	/X	Override VariableExclude

See also: [ESET](#) and [UNSET](#).

Usage:

Every program and command inherits an [environment](#), which is a list of pairs of variable **names** and **values**. Each **value** is a non-empty character string (i.e., there must be at least one character in it). Many programs use entries in the environment to modify their own actions. **TCC** itself uses several [environment variables](#).

If you simply type the SET command with no options or parameters, it will display all the names and values of all currently defined variables in the environment. Typically, you will see an entry called **PATH**, an entry called **CMDLINE**, and whatever other environment variables you and your programs have established:

```
[c:\] set
PATH=C:\;C:\UTIL
CMDLINE=C:\TCMD\TCSTART .CMD
```

If you enter only **name**, and there is no variable with that name, SET will display all environment variables whose names begin with **name**. For example, if there is no variable **pa**, the command below will display all variables whose names start with **pa**:

```
set pa
```

The above command is equivalent to the command

```
set pa*
```

If there is only a single parameter and it contains one or more wildcards (sorry, only * available), SET will display all matching environment variables. You cannot use wildcards to display the registry variables (**/D**, **/S**, **/U**, and **/V**).

You can specify variables to exclude from the SET display with the VariableExclude variable. For example, to suppress the display of the processor and user variables:

```
set VariableExclude=proc*;user*
```

(Note that this option doesn't affect the existence of the variables, just whether they're displayed by a

SET with no arguments.)

To add a variable to the environment, type SET, a space, the variable name, an equal sign, and the desired value:

```
set mine=c:\finance\myfiles
```

The variable name and the text after the equal sign will be left just as you entered it. However, case is ignored when looking for a variable; for example **MyVar**, **myvar**, and **MYVAR** all refer to the same variable. If the variable already exists, its value will be replaced with the new text that you entered.

Normally you should not put a space on either side of the equal sign. A space before the equal sign will become part of the **name**; a space after the equal sign will become part of the **value**.

Trailing whitespace in the SET command is ignored. To create a variable with trailing whitespace, use a pair of back quotes after the whitespace:

```
set mine=%@repeat[ ,20]``
```

makes **mine** 20 characters of spaces.

If you use SET to create a variable with the same name as one of the [TCC internal variables](#), you will disable the internal variable. If you later execute a batch file or alias that depends on that internal variable, it may not operate correctly. Once you delete your variable, the internal variable becomes accessible again.

To display the contents of a variable, type SET plus the variable name:

```
set mine
```

You can edit environment variables with the [ESET](#) command. To remove variables from the environment, use [UNSET](#), or type SET, followed by the variable name and an equal sign:

```
set mine=
```

The variable's **name** is limited to a maximum of 255 characters. **Name** and **value** together cannot be longer than 32,767 characters.

Note: You cannot use SET to modify [GOSUB variables](#).

The size of the environment is set automatically, and increased as necessary as you add variables.

Registry Variables

Windows stores some of its own variables in the registry. This includes Default, System, User, and Volatile variables. Those variables can be manipulated with the SET command's [/D](#), [/S](#), [/U](#) and [/V](#) options respectively. For example, to display the contents of volatile variable **clientname**, use:

```
set /v clientname
```

Note that setting a registry variable using one of the options [/D](#), [/S](#), [/U](#) or [/V](#) will not set the variable in the local environment unless you also use the [/E](#) option.

User variables are user-specific, and volatile variables are only valid for the current Windows session. Use caution when directly modifying registry variables as they may be essential to various Windows processes and applications.

If the [Update Environment on System Change](#) configuration option is set, **TCC** will monitor the WM_SETTINGCHANGE message and update the environment from the User, Volatile, and System registry entries. The update is done whenever **TCC** displays the prompt (to prevent the environment from changing in the middle of a command).

Array Variables (not in TCC/LE)

In addition to environment variables, SET is also used to set values for [array variables](#). For example, to define a 5-row by 10-column array, you would first use [SETARRAY](#):

```
setarray array1[5,10]
```

To set the array values (0-based), the syntax is:

```
set array1[a[,b[,c[,d]]]]
```

For example:

```
set array1[0,0]=Bob
set array1[0,1]=Bob's Job
```

To expand the array variable:

```
echo Name is %array1[0,0] and job is %array1[0,1]
```

Options:

- /A Evaluate the arithmetic expression on the right of the equal sign, place the result in the environment, and display it. For example, this command adds 2 and 2, and places the result in the environment variable **VAR**:

```
set /a var=2+2
```

/A interprets non numeric strings in **value** as environment variable names whether or not preceded by a percent sign %, and replaces them with their respective **values**. For example, this sequence will set **Y** to **4**:

```
set x=2
set /a y=x+2
```

You can use [@EVAL](#) to perform the same task; SET /A is included for compatibility with CMD. Unlike [@EVAL](#), use of the >> or << shift operators in SET /A requires disabling their interpretation as redirection symbols by using [SETDOS /X-6](#).

- /D Create/modify/delete a **default** variable in the registry (HKU.DEFAULT\Environment).
- /E When used together with one of [/D](#), [/S](#), [/U](#), or [/V](#), set both the registry variable and the local environment variable.

- /O Don't overwrite existing values (only valid in combination with /R).
- /P When used without a variable name, wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).

When used with a variable name and an optional prompt string, e.g. set /p **myvar**=Enter value, emulates the CMD behavior of allowing entry of a value for the variable. This is provided for compatibility reasons only. For more flexibility, use the [ESET](#) or [INPUT](#) command.
- /Q Don't echo the result of /A when at the command line.
- /R Read environment variables from a file. This is much faster than loading variables from a batch file with multiple SET commands. Each entry in the file must fit within the [command line length limit](#) for **TCC**. The file is in the same format as the SET display (*i.e.*, **name=value**), so SET /R can accept as input a file generated by redirecting SET output. For example, the following commands will save the environment variables to a file, and then reload them from that file:

```
set > varlist
set /r varlist
```

You can load variables from multiple files by listing the filenames individually after the /R.

If you are creating a SET /R file by hand, and need to create an entry that spans multiple lines in the file, you can do so by terminating each line (except the last) with an [escape character](#). However, you cannot use this method to exceed the command line length limit. You can also add comment lines to the file by starting each with a colon :. You can also use other special characters, e.g., trailing whitespace, redirection and pipe symbols (<>|), without the need for escaping the characters. If you reference the value of another variable in **value** (e.g., **x=%path;c:\jpssoft**), evaluating that variable (**path** in the example) is postponed until at some future time a command line evaluates the current variable (**x** in the example), so that the command **echo %x** will display the **path** in effect when **echo** is executed, regardless of what **path** may have been when the original SET defined **x**.

If you do not specify a filename and input is redirected, **SET /R** will read from stdin.Create a read-only variable. Once you have set the variable, you cannot change it or UNSET it. Only environment variables can be read-only, not registry variables or array variables. A read-only variable will automatically be exported from an [ENDLOCAL](#).
- /S Create/modify/delete a **system** variable in the registry (HKLM\System\CurrentControlSet\Control\Session Manager\Environment).
- /U Create/modify/delete a **user** variable in the registry (HKCU\Environment).
- /V Create/modify/delete a **volatile** variable in the registry (HKCU\Volatile Environment).
- /X Override the **VariableExclude** variable and display all matching variables.

4.3.129 SETARRAY

Purpose: Define array variables

Format: SETARRAY [/R filename arrayname] name[a[,b[,c[,d]]]] [...]

a, b, c, d Array dimensions

/R (ead)

Usage:

You can define up to 4-dimensional arrays. For example, to define a 5-row by 10-column array:

```
setarray array1[5,10]
```

The array elements are addressed in base 0, so to reference this array you would use 0-4 for the rows and 0-9 for the columns.

To set the variable elements, use the [SET](#) command.

If you don't enter any arguments, SETARRAY will display the currently defined arrays. If you don't enter any dimensions, SETARRAY will display the definition for that array. You can use wildcards in the array name.

See also [@ARRAYINFO](#).

Options:

/R Read a file into a 1-dimensional array. SETARRAY will automatically determine the required size of the array.

4.3.130 SETERROR

Purpose: Set the ERRORLEVEL value

Format: SETERROR errorlevel

errorlevel New value for ERRORLEVEL

Usage:

SETERROR sets the value of the [ERRORLEVEL](#) internal variable and the last-error code in Windows to the specified value.

See also [IF](#).

4.3.131 SETDOS

Purpose: Display or set the **TCC** configuration

Format: SETDOS [/A? /C? /D? /E? /Fn.n /G?? /I[+|-] *command* /M? /N? /P? /S?:? /V? /X[+|-]n]

/A(NSI)

/M(ode for editing)

/C(ompound)	/N(o clobber)
/D(escriptions)	/P(arameter character)
/E(scape character)	/S(hape of cursor)
/FORMAT for @EVAL)	/V(erbose)
/G (numeric separators)	/X (expansion, special characters)
/I(nternal)	

See also: [OPTION](#).

Usage:

SETDOS allows you to customize certain aspects of **TCC** to suit your personal tastes or the configuration of your system.

You can display the value of all SETDOS options by entering the SETDOS command with no parameters.

Most of the SETDOS options can also be changed in the [configuration dialogs](#). The name of the corresponding configuration option is listed with each SETDOS option below; if none is listed, that option cannot be set from the configuration dialogs. You can also define the SETDOS options in your **TCSTART** or other startup file (see [Automatic Batch Files](#)), in aliases, or at the command line.

Note: The functionality of the "/Y" option ("debug", no longer supported) of previous versions has been moved to the [BDEBUGGER](#) command.

Inheritance

When a new instance of the command is started, it inherits the SETDOS characteristics set by the most recently started instance of **TCC**.

Options:

- /A [ANSI] This option determines whether [ANSI X3.64 support](#) is enabled. /A1 enables ANSI X3.64 string processing. The default of /A0 disables ANSI X3.64 strings. See the [ANSI X3.64 Commands Reference](#) for a list of the ANSI X3.64 sequences supported by **TCC**. See also: the [ANSI Colors](#) configuration option and the [ANSI](#) internal variable.
- /C [Command Separator] This option sets the character used for separating multiple commands on the same line. The default value is the ampersand [&]. You cannot use any of the [redirection](#) characters (| > <), or a space, tab, comma, or equal sign as the command separator. The command separator is saved by [SETLOCAL](#) and restored by [ENDLOCAL](#). The following example changes the separator character to a tilde [~]:

```
setdos /c~
```

- /D [Descriptions and Description Name] This option controls whether file processing commands like [COPY](#), [DEL](#), [MOVE](#), and [REN](#) process file descriptions along with the files they belong to. /D1 turns description processing on, which is the default. /D0 turns description processing off. See also: the [Enable Descriptions](#) configuration option.

You can also use /D to set the name of the hidden file in each directory that contains file descriptions. To do so, follow /D with the filename in quotes:

```
setdos /d"files.bbs"
```

Use this option with caution, because changing the name of the description file will make it difficult to transfer file descriptions to another system.

- /E [Escape Character] This option sets the character used to suppress the normal meaning of the following character. Any character following the [escape character](#) will be passed unmodified to the command. The default escape character is a caret [[^]]. You cannot use any of the [redirection](#) characters (<|>|<|>|&|,|,|,|=) or a space, tab, comma, or equal sign as the escape character. The escape character is saved by [SETLOCAL](#) and restored by [ENDLOCAL](#). Certain characters (**b**, **c**, **e**, **f**, **k**, **n**, **q**, **r**, **s**, and **t**) have special meanings when immediately preceded by the escape character.
- /F [@EVAL maximum and minimum] This option lets you set the default decimal display precision for the [@EVAL](#) variable function. The maximum precision is 1,000 digits to the left of the decimal point and 1,000 digits to the right of the decimal point. (You can specify up to 10,000 digits in an @EVAL calculation by using the =x,y option.)

The format for this option is **/Fx.y**, where the x value sets the minimum number of digits to the right of the decimal point and the y value sets the maximum number of digits. You can use =x,y instead of =x.y if the comma is your decimal separator. Both values can range from 0 to 10. You can specify either or both values: **/F2.5**, **/F2**, and **/F.5** are all valid entries. If x is greater than y, it is ignored; if only x is specified, y is set to the same value (e.g. **/F2** is equivalent to **/F2.2**). See the [@EVAL Precision](#) configuration option to set the precision when **TCC** starts; see the [@EVAL](#) function if you want to set the display precision for a single computation.

- /G [Decimal and thousands separator characters] This option sets the [Decimal](#) and [Thousands](#) separator characters. The format is **/Gxy** where "x" is the new decimal separator and "y" is the new thousands separator. Both characters must be included. The only valid settings are **/G.**, (period is the decimal separator, comma is the thousands separator); **/G..** (the reverse); or **/G0** to remove any custom setting and use the default separators associated with your current country code (this is the default).

The decimal separator is used for [@EVAL](#), numeric [IF](#) and [IFF](#) tests, version numbers, and other similar uses. The thousands separator is used for numeric output, and is skipped when performing calculations in @EVAL.

- /I This option allows you to disable or enable internal commands. To disable a command, precede the command name with a minus [-]. To re-enable a command, precede it with a plus [+]. For example, to disable the internal LIST command to force **TCC** to use an external command:

```
setdos /i-list
```

To re-enable all disabled commands use **/I***.

- /M [Edit Mode] This option controls the initial line editing mode. To start in overstrike mode at the beginning of each command line, use **/M0** (the default). To start in insert mode, use **/M1**. See also: the [Edit Mode](#) configuration option.
- /N [NoClobber] This option controls output [redirection](#). **/N0** means existing files will be overwritten by output redirection (with >) and that appending (with >>) does not require the

file to exist already. This is the default. **/N1** means existing files may not be overwritten by output redirection, and that when appending the output file must exist. A **/N1** setting can be overridden with the [!] character. See also: the [Protect Redirected Output File](#) configuration option.

/P [Parameter Character] This option sets the character used after a percent sign to specify all or all remaining command line parameters in a [batch file](#) or [alias](#). The default value is the dollar sign [\$]. The parameter character is saved by [SETLOCAL](#) and restored by [ENDLOCAL](#).

/S [Insert and Overstrike Cursor] The cursor size is entered as a percentage of the total character height. The default values are 10:100 (a 10% underscore cursor for overstrike mode, and a 100% block cursor for insert mode). Because of the way video drivers remap the cursor shape, you may not get a smooth progression in the cursor size from 1% - 100%. (You can disable the cursor by specifying a size of 0:0.)

If either value is -1, **TCC** will not attempt to modify the cursor shape at all. You can retrieve the current cursor shape values with the [%_CI](#) and [%_CO](#) internal variables. See also the [Overstrike Cursor](#) and [Insert Cursor](#) configuration options.

/V [Batch Echo] This option controls the default for command echoing in batch files.

/V0 disables echoing of batch file commands unless [ECHO](#) is explicitly set ON.

/V1, the default setting, enables echoing of batch file commands unless [ECHO](#) is explicitly set OFF. See also: the [Batch Echo](#) configuration option.

/X[+|-]n (expansion and special characters) This option enables and disables alias and environment variable expansion, and controls whether special characters have their usual meaning or are treated as text. It is most often used in batch files to process text strings which may contain special characters.

The features enabled or disabled by **/X** are numbered (in hex). All features are enabled when **TCC** starts, and you can re-enable all features at any time by using **/X0**. To disable a particular feature, use **/X-n**, where **n** is the feature number from the list below. To re-enable the feature, use **/X+n**. To enable or disable multiple individual features, list their numbers in sequence after the + or - (e.g. **/X-345** to disable features 3, 4, and 5).

The features are:

- 1** All alias expansion
- 2** Nested alias expansion only
- 3** All variable expansion (includes environment variables, batch file parameters, variable function evaluation, and alias parameters)
- 4** Nested variable expansion only
- 5** Multiple commands, conditional commands, and piping (affects the command separator, ||, &&, |, and |&)
- 6** Redirection (affects < , >, >&, >&>, etc.)
- 7** Quoting (affects back-quotes ` and double quotes ") and square brackets)
- 8** Escape character
- 9** [Include lists](#)
- A** [User-defined functions](#)

If nested alias expansion is disabled (/X-2), the first alias of a command is expanded but any aliases it invokes are not expanded. If nested variable expansion is disabled (X-4), each variable is expanded once, but variables containing the names of other variables are not expanded further.

For example, to disable all features except alias expansion while you are processing a text file containing special characters:

```
setdos /x-35678
      ... [perform text processing here]
setdos /x0
```

A [SETLOCAL](#) command will save the current SETDOS /X values for [ENDLOCAL](#) to restore.

4.3.132 SETLOCAL

Purpose: Save a copy of the current disk drive, directory, environment, alias and function lists, and special characters

Format: SETLOCAL

See also: [ENDLOCAL](#).

Usage:

SETLOCAL can be used on the command line, in aliases, and in batch files.

SETLOCAL will save :

- the default disk drive and directory
- the environment,
- the alias list
- the user-defined function list (not in **TCC/LE**)
- the special character set (command separator, escape character, parameter character, decimal separator, and thousands separator)
- the [SETDOS /X](#) setting

After using SETLOCAL, you can change the values of any or all of the above, and later restore the original values with an [ENDLOCAL](#) command, or just by exiting the batch file.

SETLOCAL does not save the command history or array variables.

For example, this batch file fragment saves everything, removes all aliases so that aliases will not affect batch file commands, changes the disk and directory, changes the command separator, runs a program, and then restores the original values:

```
setlocal
unalias *
cdd d:\test
setdos /c~
program ~ echo Done!
endlocal
```

SETLOCAL and ENDLOCAL may be nested up to 16 levels deep in each batch file. You can also have multiple SETLOCAL / [ENDLOCAL](#) pairs within a batch file, and nested batch files can each have their own SETLOCAL / [ENDLOCAL](#) pairs.

SETLOCAL does not override the [Local Aliases](#) configuration option. Consequently changing aliases inside a SETLOCAL / [ENDLOCAL](#) pair affects the definition of aliases of other concurrently executing sessions of **TCC**.

You can also use SETLOCAL and [ENDLOCAL](#) in an alias or at the command line. The maximum nesting level from a command line or alias is 10 levels. Unlike batch files, you are responsible for matching the SETLOCAL / [ENDLOCAL](#) calls from an alias or command line; **TCC** will not perform an automatic ENDLOCAL.

An ENDLOCAL is performed automatically at the end of a batch file, or when returning from a "[GOSUB](#) filename". If you invoke one batch file from another without using [CALL](#), the first batch file is terminated, and an automatic [ENDLOCAL](#) is performed; the second batch file inherits the settings as they were prior to any SETLOCAL.

You can "export" modified variables from inside a SETLOCAL / ENDLOCAL block. See [ENDLOCAL](#) for details.

4.3.133 SHIFT

Purpose: Allows the use of more than 512 parameters in a batch file, or iterating through its parameters

Format: SHIFT [[-]n | /n]

n Number of positions to shift (an unsigned number), or the position of the parameter to be deleted.

Usage:

SHIFT is provided for compatibility with batch files written for CMD, where it was used to access more than the CMD limit of 10 parameters. **TCC** supports 4096 parameters (%0 to %4095), so you do not need to use SHIFT for batch files running exclusively under **TCC**.

SHIFT *n* moves each of the batch file parameters *n* positions to the left. The default value for *n* is 1. For example, SHIFT (with no parameters) makes the parameter %1 become to %0, the parameter %2 becomes %1, etc.

SHIFT -*n* moves parameters to the right, but it is limited to moving them back to their position on entry to the batch file.

This form of SHIFT also affects the special parameters %n\$, %% and %# (number of command parameters). However, for compatibility with CMD, this form of the SHIFT command does not alter the contents or order of the parameters returned by %. See [Batch File Parameters](#) for details.

For example, create a batch file called TEST.BAT:

```
echo %1 %2 %3 %4
shift
echo %1 %2 %3 %4
```

```
shift 2
echo %1 %2 %3 %4
shift -1
echo %1 %2 %3 %4
```

Executing the command below produces the following results:

```
[c:\] test one two three four five six seven
one two three four
two three four five
four five six seven
three four five six
```

SHIFT /n This form of the command irreversibly deletes parameter **%n** from the command tail, and shifts all parameters originally to its right 1 position to the left. For example,

```
shift /2
```

leaves parameters **%0** and **%1** unchanged, and moves the value of **%3** to position **%2**, **%4** to **%3**, etc.

This form of SHIFT also affects the special parameters **%n\$**, **%%** and **%#** (number of batch file parameters), and unlike the first form, it also affects **%***. See [Batch File Parameters](#) for details.

4.3.134 SHORTCUT

Purpose: Create or display a shortcut

Format: [Creation mode](#)

SHORTCUT command args dir desc link mode [iconfile [iconoffset [hotkey]]]

[Display mode](#)

SHORTCUT link

command	Command the shortcut executes
args	Command line parameters for command
dir	Starting directory
desc	Description
link	Filename of the .LNK file.
mode	Initial window mode: 1=normal, 2=minimized, 3=maximized
iconfile	File containing the icon to use
iconoffset	Icon offset within iconfile
hotkey	Hotkey to invoke the shortcut

Usage:

Creation Mode

SHORTCUT creates a Windows shortcut file and places it in the specified directory. You can run any Windows shortcut from **TCC** by entering the name of the .LNK file on the command line.

SHORTCUT requires a minimum of 6 parameters. To leave a parameter blank, enter an empty string (2

double quotes "" in its place. Any parameter must be enclosed in double quotes if it includes white space or other special characters.

Command is the full path of the executable file to start, or the data file or folder to open. If it is a data file, its extension must be associated with an executable command (see [ASSOC](#)) for the shortcut to work.

The **args** parameter lists any command line parameters which you want to include when **command** is executed. For example, if **command** points to a batch file, you might want to include /c in **args** so that **TCC** exits immediately when the batch file is completed.

The **dir** parameter is the path of the directory to which you want Windows to switch when the command starts. If you don't care which directory is used, you can omit this parameter by entering "" in its place.

Desc provides a description that is stored internally in the shortcut. It is displayed when the cursor is moved to the shortcut. If you omit the description, enter "" in its place.

The **link** parameter is the drive, path, name and extension of the shortcut file you want to create. The drive and path portion is interpreted according to the usual rules - missing elements default to the current defaults, path is relative to the current default unless it starts with \. The file extension must be .LNK.

Note: If you want the shortcut to appear on the Windows desktop, you should include the full path to one of the desktop directory in the command. In most Windows configurations, that directory can be referenced symbolically as %userprofile\Desktop. Some Windows versions also include an All Users \Desktop directory.

The **mode** parameter determines how Windows will display the application or folder when you run the shortcut. It must be 1 for a normal window, 2 for a minimized window (normally placed on the taskbar), or 3 for a maximized window.

The two (optional) parameters, **iconfile** and **iconoffset** allow you to specify the icon for the shortcut to use. (By default, SHORTCUT will use the default icon in the executable file.)

The final (optional) parameter **hotkey** specifies the keystroke which will call the shortcut. The keystroke should be entered in the same format as used in [KEYSTACK](#); for example, **Ctrl-Alt-B**.

Display mode

If you provide a single parameter (a link file name), SHORTCUT will display the values for that link.

4.3.135 SHRALIAS

Purpose: Retains global command history, directory history, alias and user function lists in memory when **TCC** is not running

Format: SHRALIAS [/U]

[/U\(nload\)](#)

Usage:

When you close all **TCC** sessions, the memory for the global command history, global directory history, global alias and global function lists is released. If you want the lists to be retained in memory even

when **TCC** is not running, you need to execute SHRALIAS.

The SHRALIAS command starts and initializes SHRALIAS.EXE, a small program which remains active and retains global lists when **TCC** is not running. SHRALIAS.EXE must be stored in the same directory as **TCC** or in a directory on your PATH. You cannot run SHRALIAS.EXE directly, it must be invoked internally by the SHRALIAS command.

Once SHRALIAS has been executed, the global lists will be retained in memory until you use SHRALIAS /U to unload the lists, or until you shut down your operating system.

If you have an environment variable named SHRALIAS_SAVE_PATH, SHRALIAS will save the alias, history, dirhistory, and function lists to the path specified by SHRALIAS_SAVE_PATH when SHRALIAS exits. The files will be saved in Unicode format as alias.sav, history.sav, dirhistory.sav, and function.sav.

SHRALIAS will not work unless you have at least one copy of **TCC** running with global alias, global function, global command history, or global directory history enabled. If no global list is found, SHRALIAS will display an error.

If you start SHRALIAS from a temporary **TCC** session which exits after starting SHRALIAS, the **TCC** session may terminate and discard the shared lists before SHRALIAS can attach to them. In this case SHRALIAS.EXE will not be loaded. If you experience this problem, add a short delay with the [DELAY](#) command after SHRALIAS is loaded and before your session exits.

SHRALIAS will not work in detached sessions (i.e., those started with [DETACH](#), or with the AT utility), due to security issues within Windows. Therefore the SHRALIAS command is ignored for detached sessions.

For more information about global histories, function and alias lists, see [Local and Global History Lists](#), [Local and Global Functions](#), [Local and Global Aliases](#).

Option:

- /U** Shuts down SHRALIAS.EXE. All global command history, directory history, function and alias lists will be released from memory when the last copy of **TCC** exits unless SHRALIAS is loaded again before that time.

4.3.136 SMPP

Purpose: Send simple text (**SMS**) messages, typically to text-enabled cellular phones and similar devices

Format: SMPP server username password recipient message

server	SMS server name
username	User name for the SMS server
password	Password for the SMS server
recipient	Phone number or dotted IP of an SMS-enabled device
message	The message to send

See also: [SENDMAIL](#), [SNPP](#).

Usage:

SMPP sends **message** through standard Internet Paging Gateways. Depending on your system configuration, you may need to start an Internet connection before using SMPP. See your service provider for specific requirements.

4.3.137 SNMP

Purpose: Send SNMP traps

Format: SNMP remotehost trapOID "value" [username password]

remotehost	Host name receiving the trap
trapOID	OID of the trap
value	Description
username	User name for SNMP v3 trap
password	Password for SNMP v3 trap

Usage:

SNMP normally sends an SNMPv2 trap. If you specify a user name and password it will send an SNMPv3 trap.

The following symbolic names are recognized and translated:

Trap Name	OID
coldStart	1.3.6.1.6.3.1.1.5.1
warmStart	1.3.6.1.6.3.1.1.5.2
linkDown	1.3.6.1.6.3.1.1.5.3
linkUp	1.3.6.1.6.3.1.1.5.4
authenticationFailure	1.3.6.1.6.3.1.1.5.5
egpNeighborLoss	1.3.6.1.6.3.1.1.5.6
enterpriseSpecific	1.3.6.1.6.3.1.1.5.7

4.3.138 SNPP

Purpose: Send messages to alphanumeric pagers

Format SNPP server pagerid message

server	The SNPP server name
pagerid	The ID of the pager to receive the message
message	The message to send

See also: [SENDMAIL](#), [SMPP](#).

Usage:

SNPP sends **message** to alphanumeric pagers through standard Internet Paging Gateways. Depending on your system configuration, you may need to start an Internet connection before using SNPP.

4.3.139 START

Purpose: Start a program in another session or window

Format: START ["*title*"] [/AFFINITY=n /ABOVENORMAL /BELOWNORMAL /DESKTOP=name /ELEVATED /HIGH /LOW /NORMAL /REALTIME /B /C /K /Dpath /I /FS /INV /MAX /MIN /NODE n /POS=x,y,width,height /L /LA /LD /LF /LH /MONITOR=n /RUNAS user password /SEPARATE /SHARED /SIZE=rows,cols /TAB /TABNA /WAIT /WIN /PGM] "*progname*" [*command*]

<i>title</i>	Title to appear on title bar
<i>path</i>	Startup directory
<i>progname</i>	Program name (not the session name)
<i>command</i>	Command to be executed by <i>progname</i>

/ABOVENORMAL	Priority	/LOW	Priority
/AFFINITY	Multiple CPUs	/MAX	Maximized window
/B	No new console	/MIN	Minimized window
/BELOWNORMAL	Priority	/MONITOR	Monitor to use
/C	Close when done	/NODE	NUMA node
/D	Startup directory	/NORMAL	Priority
/DESKTOP	Start desktop	/PGM	Program name
/ELEVATED	Start as admin	/POS	Position of window
/FS	Full screen window	/REALTIME	Priority
/HIGH	Priority	/RUNAS	Run as other user
/I	Inherit environment	/SEPARATE	Separate session
/INV	Invisible window	/SHARED	Shared session
/K	Keep when done	/SIZE	Screen buffer size
/L	Local lists	/TAB	Start in Take Command tab window
/LA	Local aliases	/TABNA	Start in inactive Take Command tab
/LD	Local directory history	/WAIT	For session to finish
/LF	Local functions	/WIN	Windowed session
/LH	Local history list		

See also: [DETACH](#).

Usage:

START is used to begin a new session, and optionally run a program in that session. If you use START with no parameters, it will begin a new **TCC** session. If you add a **command**, START will begin a new session or window and execute that command.

START will return to the **TCC** prompt immediately (or continue a batch file), without waiting for the program to complete, unless you use [/WAIT](#).

If ***title*** is included, it will appear on the task list and **Alt-Tab** displays instead of the program name. ***Title*** must be enclosed in double quotes, and cannot exceed 127 characters.

START always assumes that the first quoted string on the command line is the ***title***. If there is a second quoted string it is assumed to be the **command**. As a result, if the name of the program you are starting contains white space (and must therefore be quoted), and you don't specify a ***title***, START will interpret

the first quoted string as the **title**, not the **command**. To address this, use the [/PGM](#) switch to indicate explicitly that the quoted string is the program name, or include a title before the program name. For example, to start the program C:\Program Files\Proc.Exe you could use either of the first two commands below, but the third command would not work:

Valid

```
start /PGM "C:\Program Files\Proc.Exe"  
start "test" "C:\Program Files\Proc.Exe"
```

Invalid

```
start "C:\Program Files\Proc.Exe"
```

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

START offers a large number of switches to control the session you start. In most cases you need only a few switches to accomplish what you want. The list below summarizes the most commonly used START options, and how you can use them to control the way a session is started.

Window controls: [/FS](#), [/MAX](#), [/MIN](#), and [/POS](#) allow you to start a character-mode windowed session in full screen mode, a maximized window, a minimized window, or a window with a specified position and size, respectively. [/INV](#) starts an invisible window. [/B](#) starts the program in the current console window. The default is [/WIN](#), which permits Windows to choose the position and size of the non-maximized window. If you start a graphics mode program, only [/MAX](#) and [/POS](#) are effective, and the position and size information associated with [/POS](#) is ignored. Windows will use the size, but not the position of the same program when last used in **RESTORE** mode. If you want to control the window size and placement of a graphics mode program, use the [ACTIVATE](#) command after the window has been opened.

Session priority: The options [/ABOVENORMAL](#), [/BELOWNORMAL](#), [/HIGH](#), [/LOW](#), [/NORMAL](#) and [/REALTIME](#) allow you to select the new session's priority.

Program controls.

If **progname** is in the "App Paths" registry, its associated "Path" value (if it exists) is inserted into the beginning of the [PATH](#) in the environment inherited by the program.

If **progname** is the name of a directory instead of an executable program, **TCC** will start your default Windows shell (usually Windows Explorer) in the specified directory.

Progname inherits the environment as it exists when START is executed, unless [/I](#) is used to select the default environment.

If **progname** specifies **TCC.EXE**, the options [/L](#), [/LA](#), [/LD](#), [/LF](#) and [/LH](#) provide control over the use of local or global lists. See details below.

The initial directory for **progname** is the current default directory, unless otherwise specified using the [/D](#) option.

If **progname** is a 16-bit Windows application, by default it starts in a shared virtual machine. You may use the [/SEPARATE](#) option to force creation of a unique virtual machine.

When **command** is finished, [/C](#) closes the session (the default for Windows sessions), while [/K](#) keeps it and displays the prompt (the default for character mode sessions).

The Process ID of the detached session or program is returned in the [_STARTPID](#) internal variable.

Options:

/=	Display the START command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in TCC/LE .)
/ABOVENORMAL	Set the priority above normal.
/AFFINITY=n	On multiple processor machines, set the processor affinity for this process. /AFFINITY takes a hex argument for the processor mask -- i.e., to set the affinity for cpu's 1 and 4, set /AFFINITY=5.
/B	The program is started without creating a new window or console, i.e. in the TCC window. Normally, the application is started in its own window. For compatibility with CMD, /B also disables Ctrl-C processing for the program.
/BELOWNORMAL	Set the priority below normal.
/C	Start the program in a new TCC window and close the TCC window when the application ends.
/D	Specifies the startup directory. Include the directory name immediately after the /D, with no intervening spaces or punctuation.
/DESKTOP	Start the program on the specified desktop
/ELEVATED	Start the program elevated, with full admin privileges. (Windows Vista or later only.) /ELEVATED cannot be used in combination with /RUNAS.
/FS	Start the console application in full-screen mode. (Not supported in Windows Vista or later.)
/HIGH	Start the window at high priority.
/I	Inherit the default (startup) environment, rather than the current environment. (Only supported in Vista and later.)
/INV	Start the session or window as invisible. No icon will appear and the session will only be accessible through the Task Manager or Window List.
/K	Start the program in a new TCC window and keep the TCC window open when the program ends. (Use the EXIT command to close the TCC window.)

/L	Start TCC with local alias, function, history and directory history lists. This option is equivalent to specifying all of /LA, /LD, /LF, and /LH (below).
/LA	Start TCC with a local alias list. See ALIAS for information on local and global alias lists.
/LD	Start TCC with a local directory history list. See Local and Global History Lists for information on local and global directory history lists.
/LF	Start TCC with a local function list. See FUNCTION for information on local and global function lists.
/LH	Start TCC with a local history list. See Local and Global History Lists for information on local and global history lists.
/LOW	Start the window at low priority.
/MAX	Start the session or window maximized.
/MIN	Start the session or window minimized.
/MONITOR=n	Start the program on the specified monitor (1 to n). This will only work with apps that do not try to position their window at startup, and you cannot combine this switch with /POS.
/NODE n	Start the program using the specified NUMA node (<i>n</i> is a decimal integer).
/NORMAL	Start the window at normal priority.
/PGM	The quoted string following this option is the program name. Any additional text beyond the quoted string is passed to the program as its parameters, so to use other START switches you must place them before /PGM which must be the last option for START. You can use /PGM to allow START to differentiate between a quoted long filename and a quoted title for the session.
/POS=left,top,width,height	Start the window at the specified screen position. The top left corner of the screen is 0,0.
/REALTIME	Start the window at realtime priority.
/RUNAS	Run a command in the context of the specified user. The syntax is: /RUNAS user@domain password . If "domain" is not specified, the local database is checked for the username. If you specify * for the password, START will prompt you to enter the password. (Useful when you don't

	want to put the password in a batch file.) /RUNAS cannot be used in combination with /ELEVATED.
/SEPARATE	Start a 16-bit Windows application in a separate virtual machine. Normally, all 16-bit Windows applications are started in the same virtual machine, see /SHARED..
/SHARED	Start a 16-bit Windows application in the shared virtual machine (default). See also /SEPARATE. Included only for compatibility with CMD.
/SIZE=rows,columns	Specifies the screen buffer size. Rows is the number of text rows and columns is the number of text columns. (This is not the size of the session's window.)
/TAB	Start the command in a new TCC tab window, and activate the new window. The command will usually be a Windows console mode application, but Take Command can also run many simple GUI applications in a tab window (provided the application does not have multiple parent windows).
/TABNA	Start the command in a new TCC tab window, but don't activate the new window.
/WAIT	Wait for the new session or window to finish before continuing.
/WIN	Start the new console session as a window (this is the default.) See also /FS and /B .

4.3.140 STATUSBAR

Purpose: Display a message on the **Take Command** status bar

Format: STATUSBAR message

message Text to display.

Usage:

STATUSBAR parses and expands *message*, and displays it on the **Take Command** status bar.

4.3.141 SWITCH

Purpose: Select commands to execute in a batch file based on a value

Format: SWITCH expression
CASE value1 [.OR. value2 [.OR. value3 ...]]
[commands]
CASE value4
[commands]
CASEALL

```
[commands]
[DEFAULT
  commands]
ENDSWITCH
```

expression An environment variable, internal variable, variable function, text string, or a combination of these elements, that is used to select a group of commands.

value1, value2 A value to test or multiple values connected with .OR.
commands One or more commands to execute if the expression matches the value. If you use multiple commands, they must be separated by command separators or placed on separate lines of a batch file.

See also: [IF](#) and [IFF](#).

Usage:

SWITCH can only be used in batch files. It allows you to select a command or group of commands to execute based on the possible values of a variable or a combination of variables and text.

The SWITCH command is always followed by an **expression** created from environment variables, internal variables, variable functions, and text strings, and then by a sequence of CASE statements matching the possible **values** of **expression**, an optional DEFAULT statement, and terminated by an ENDSWITCH statement. Each CASE statement and the DEFAULT statement may be followed by one or more **commands**.

TCC evaluates **expression**, and sequentially compares it with the list of **values** in the CASE statements, starting with the first one. Comparison rules are the same ones used for the **EQ** relational operator; see [Numerical and String Comparisons](#) for details. If a match is found, the **commands** following the matched CASE statement are executed, and the batch file continues with the commands that follow ENDSWITCH. If there are any matches in subsequent CASE statements, they are ignored. The **value** in a CASE statement can be literals, or variables or functions (which will be expanded prior to the comparison with the SWITCH expression).

The optional CASEALL statement should follow all of the CASE statements but precede DEFAULT. If any preceding CASE block was executed, CASEALL will also be executed; otherwise it is ignored.

If during the search for a match the DEFAULT statement is encountered, the **commands**, if any, following it are executed, and the batch file continues with the commands that follow ENDSWITCH. Any CASE statements after the DEFAULT statement are ignored.

SWITCH commands can be nested.

You can exit from all SWITCH / ENDSWITCH processing by using [GOTO](#) to a line past the last ENDSWITCH.

Restrictions

Each SWITCH, CASE, DEFAULT and ENDSWITCH statement must be on a separate line, and may not be followed by a command separator. (This is the reason SWITCH cannot be used in aliases.) There is no restriction on grouping and command separator use in the **commands** for a CASE or DEFAULT.

You can link a list of values in a single CASE statement with .OR., but not with .AND. or .XOR..

Examples

The batch file fragment below displays one message if the user presses **A**, another if the user presses **B** or **C**, and a third one if the user presses any other key:

```
inkey Enter a keystroke: %%key
switch %key
case A
    echo It's an A
case B .or. C
    echo It's either B or C
default
    echo It's none of A, B, or C
endswitch
```

In the example above, the value of a single environment variable was used for **expression**. However, you can use other kinds of expressions if necessary. The first SWITCH statement below selects a command to execute based on the length of a variable, and the second bases the action on a quoted text string stored in an environment variable:

```
switch %@len[%var1]
case 0
    echo Missing var1
case 1
    echo Single character
...
endswitch

switch "%string1"
case "This is a test"
    echo Test string
case "The quick brown fox"
    echo It's the fox
...
endswitch
```

4.3.142 SYNC

Purpose: Synchronize two directories

Format: SYNC [/A:... /C /D /E /F /G /J /K /L /M /N[enst] /O /O:[-]adegrnstu /P /Q /R /S[[+]n] /T /V / W /X/Y /Z] dir1 dir2

dir1	First directory (and source for a /W)
dir2	Second directory (and target for a /W)

/A:...	Attribute switch	/O	Only if no target file
/C.	Changed source files	/	Sort order
/D	Copy encrypted files	O:...	
/E	No error messages	/P	Prompt
/F	No empty subdirectories	/Q	Quiet
		/R	Replace

/G	Display percentage completed	/S	Subdirectories included
/H	H(idden included)	/T	Totals
/I"text"	Match description	/V	Verify
/J	Restartable copy	/W	Delete non-matching target
/K	Keep RDONLY attribute	/X	Clear archive bit
/L	ASCII-mode FTP transfer	/Y	Suppress prompt
/M	Modified files (not Archived)	/Z	Overwrite read-only
/N	Disable		

See also: [COPY](#) and [MOVE](#).

File Selection

Supports [command dialog](#), extended [wildcards](#) and [ranges](#).

Internet: Can be used with [FTP servers](#).

Usage:

SYNC will synchronize two directories, copying the updated files from each directory to the other. If you don't specify any arguments, SYNC will display its command dialog.

SYNC sets three internal variables:

%_sync_dirs	The number of directories created
%_sync_files	The number of files copied
%_sync_errors	The number of errors

Options:

- /= Display the SYNC command dialog to help you set the directory and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A:... Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. See the cautionary note under **Advanced Features** above before using /A: when both **dir1** and **dir2** contain file descriptions. Hidden or system files selected by this option overwrite hidden or system files in the target directory.

You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /C Copy files only if the destination file exists and is older than the source file. This option is useful for updating the files in one directory from those in another without copying any files not already in the target directory.
- /D Force copy of an encrypted file even when the target will be decrypted.
- /E Suppress all non-fatal error messages, such as **File not found** or **Can't copy file to itself**. Fatal error messages, such as **Drive not ready**, will still be displayed. This option is most useful in batch files and aliases.
- /F When used with /S, SYNC will not create any empty subdirectories.

- /G** Displays the percentage copied, the transfer rate (in Kbytes/second), and the estimated time remaining. Useful when copying large files across a network or via FTP to ensure the copy is proceeding. When [/V](#) is also used, reports percentage verified.
- /H** Copy all matching files including those with the hidden and/or system attribute set. See the cautionary note under **Advanced Features** above before using /H when both **dir1** and **dir2** contain file descriptions.
- /I"text"** Select source files by matching text in their descriptions. See [Description Ranges](#) for details.
- /J** Copy the files in restartable mode. The copy progress is tracked in the destination file in case the copy fails. The copy can be restarted by specifying the same source and destination file names.
- /K** (Keep read-only attribute) SYNC normally maintains the hidden and system attributes, sets the archive attribute, and removes the read-only attribute on the target file. /K tells SYNC to also maintain the read-only attribute on the **destination** file.
- /L** Perform FTP transfers in ASCII mode, instead of the default binary mode.
- /M** Copy only those files with the archive attribute set, *i.e.*, those which have been modified since the last backup. The archive attribute of the source file will not be cleared after copying; to clear it use the [/X](#) switch, or use [ATTRIB](#).
- /N** Do everything except actually perform the copy. This option is useful for testing the result of a complex SYNC command. /N displays how many files would be copied. /N does not prevent creation of destination subdirectories when it is used with [/S](#).

A **/N** with one or more of the following arguments has an alternate meaning:

- d** Skip hidden directories (when used with [/S](#))
- e** Don't display errors
- j** Skip junctions (when used with [/S](#))
- n** Don't update the file descriptions
- s** Don't display the summary
- t** Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*)

- /O:...** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see [/T:acw](#)
- e** Sort by extension
- g** Group subdirectories first, then files

- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

- /P** Ask the user to confirm each source file. Your options at the prompt are explained in detail under [Page and File Prompts](#). See also: the [/Q](#) option below.
- /Q** Don't display filenames, percentage copied, total number of files copied, etc... When used in combination with the [/P](#) option above, it will prompt for filenames but will not display the totals. This option is most often used in batch files. See also [/T](#).
- /R** Prompt the user before overwriting an existing file. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /S** Copy the subdirectory tree starting with the files in the source directory plus each subdirectory below that. If the destination subdirectories don't exist, SYNC will attempt to create them. If SYNC /S creates one or more destination directories, they will be added automatically to the [extended directory search](#) database.

If you attempt to use SYNC /S to copy a subdirectory tree into part of itself, SYNC will detect the resulting infinite loop, display an error message and exit.

If you specify a number after the /S, SYNC will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\etc", /S2 will only affect the "a", "b", and "c" directories.

If you specify a + followed by a number after the /S, SYNC will not sync any files until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\etc, /S+2 will not sync anything in \a or \ab.
- /T** Turns off the display of filenames, like [/Q](#), but does display the total number of files copied.
- /V** Verify each disk write by performing a true byte-by-byte comparison between the source and the newly-created target file. This option will significantly increase the time necessary to complete a SYNC command.
- /W** Delete files in **dir2** that do not exist in **dir1**.
- /X** Clear the archive attribute from the source file after a successful copy.
- /Y** If you have the [COPY Prompt on Overwrite](#) option set, you can suppress the prompt with /Y.
- /Z** Overwrite **destination** files regardless of their attributes. Without this option, SYNC will fail with an "Access denied error" if the **destination** file has its read-only attribute set, or (depending on other options) its hidden or system attribute set. Required to overwrite read-only targets regardless of other options. Required to overwrite hidden or system targets unless the source also has the attribute, and either [/H](#) or [/A:](#) is used to select it.

4.3.143 TAIL

Purpose: Display the end of the specified file(s)

Format: TAIL [range ... [I"text"]] [A:attrlist] /B Cnn F N+x N[]n /O:[-]adegrstu P Q V {@file} file{...}

file The file or list of files that you want to display

@file A text file containing the name of a file to display in each line (see [@file lists](#) for details)

/A: (Attribute select) /N(umber of lines)

/B(ell) /O:... (Order)

/C (number of bytes) /P(ause)

/F(follow) /Q(uiet)

/I"text" (description range) /V(erbose)

/N+x (skip x lines before display)

See also: [HEAD](#), [LIST](#), and [TYPE](#).

File Selection

Supports [command dialog](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

Internet: Can be used with [FTP servers](#), including HTTP/HTTPS files, e.g.

```
tail "http://jpsoft.com/notfound.htm"
```

Usage:

The TAIL command displays the last part of a file or files. It is normally only useful for displaying ASCII text files (i.e. alphanumeric characters arranged in lines separated by CR/LF). Executable files (.EXE) and many data files may be unreadable when displayed with TAIL because they include non-alphanumeric characters or unusual line separators.

You can press **Ctrl-S** to pause TAIL's display and then any key to continue.

The following example displays the last 15 lines of the files *MEMO1* and *MEMO2*:

```
tail /n15 memo1 memo2
```

To display text from the clipboard use **CLIP:** as the file name. **CLIP:** will not return any data if the clipboard does not contain text. See [Highlighting and Copying Text](#) for additional information on **CLIP:**

TAIL sets two internal variables:

%_tail_files The number of files displayed

%_tail_errors The number of errors

- **FTP Usage**

TAIL can also display files on [FTP servers](#). For example:

```
tail "ftp://ftp.microsoft.com/index"
```

You can also use the [IFTP](#) command to start an FTP session on a server, and then use an abbreviated syntax to specify the files and directories you want.

- ***NTFS File Streams***

TAIL supports file streams on NTFS drives. You can type an individual stream by specifying the stream name, for example:

```
tail streamfile:s1
```

- ***Pipes***

TAIL can optionally be used with an input [pipe](#). For example:

```
dir | tail /n2
```

This is not ordinarily feasible in Windows because pipes can't be "rewound", and therefore the pipe has to be written to a temporary memory buffer and the TAIL taken from there. Consequently, this limits the amount you can actually display in TAIL to less than a million bytes when the input is piped.

Examples

tail /n 5 xxx	displays the last 5 lines of file xxx
tail /n+20 /n 999999 xxx	skip 20 lines, then display 999999 lines of xxx
tail /n+1001 /n 1 xxx	skip 1001 lines, then display 1 line of xxx
set x=%@execstr[tail /n+1001 /n 1 xxx]	sets x to the contents of the 1002-nd line of xxx
set x=%@execstr[tail /n 2 xxx]	sets x to the contents of the penultimate line of xxx

Options:

/= Display the TAIL command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

/A:[attributelist]

Select only those files that match the specified attribute(s). See [Attribute Switches](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

/B Ignore bell (ASCII 7) characters.

/Cnn[b|k|m]

Display **nn** bytes, 512-byte **blocks**, **kilobytes**, or **megabytes**.

/F Continuously monitor the file and display new lines until the command is interrupted, e.g.

using ***Ctrl-C*** or ***Ctrl-Break..***

/I"*text*"

Select files by a [descriptor range](#). See the link for details.

/N *n* Display *n* lines. The default is **10**. Space between the option switch /N and the number *n* is optional. If /N is specified without *n*, it is equivalent to specifying 0 lines to be displayed, and the command will not generate output, unless [/V](#) is also specified.

/N+x Skip *x* lines from the beginning of the file, then start displaying lines. If the /N+ option is specified without specifying *x*, the option is ignored. This option does not affect the number of lines displayed (unless the start line is too close to the end of file)

Example: **TAIL /N+5 *file*** will display 10 lines (the default) after skipping 5 lines.

/O:... Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see [/T:acw](#)
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

/P Pause and prompt after displaying each page.

/Q Do not display a header for each file. This is the default behavior, but an explicit /Q may be needed to override an alias that forces [/V](#).

/V Display a header for each file.

4.3.144 TAR

Purpose: Add, update, or delete files in a .tar archive

Format: **TAR [/A:[[-][+]rhsdaecjot] /A /C /D /F /G /M /O:[-]adegrstu /Q /R /TEST /U /V] tararchive
[@*file*] *file*...**

tararchive The tar file to work with

file The files(s) to be added to the tar archive

[/A:... \(attribute switch\)](#)

[/A\(dd\)](#)

[/O:... \(sort order\)](#)

[/P\(rogress\)](#)

/C(ontents)	/R(ecurse)
/D(elete)	/TEST
/F(reshen)	/U(pdate)
/G(zip)	/V(iew)
/M(ove)	

See also [UNTAR](#).

File Selection

Supports [command dialog](#), [attribute switches](#), [extended wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

Usage:

TAR is compatible with archives created by the Linux / UNIX tar utility. Unless you use the [/G](#) option, the tar file will be uncompressed. If you don't specify any arguments, TAR will display its command dialog.

You can specify a pathname for `tararchive`. If you don't provide an extension, and the filename as entered doesn't exist, TAR adds ".tar". If you don't specify an operation, TAR will default to Add.

TAR sets two internal variables:

<code>%_tar_files</code>	The number of files archived
<code>%_tar_errors</code>	The number of errors

Option:

- /=** Display the TAR command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A:....** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow `/A:..`. Do not use `/A:` with [@file lists](#). See [@file lists](#) for details.

You can specify `/A:=` to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /A** Add the specified file(s) to the tar file. (This is the default.)
- /C** Display (on standard output) the contents of a file in the tar archive.
- /D** Delete the specified file(s) from the tar file.
- /F** Update only those files that currently exist in the tar file, and which are older than the files on disk.
- /G** When all the files have been added to the archive, compress the entire archive using gzip compression and create a .tar.gz archive.
- /M** Delete the files from the disk after adding them to the tar file.
- /O:....** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the files will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

- /P** Display the progress (0 - 100%) for each file as it is archived.
- /Q** Don't display the files being archived.
- /R** If the argument is a subdirectory, copy all of the files in that subdirectory and all of its subdirectories to the tar archive.
- /TEST** Test the integrity of the TAR file (header and contents). Any errors will be displayed on STDERR.
- /U** Update files which either don't exist in the tar, or which are older than the files on disk.
- /V** View the list of files in the tar file (date, time, size, and filename).

4.3.145 TASKBAR

Purpose: Call Windows Taskbar functions

Format: TASKBAR command

Usage:

TASKBAR calls the Windows Taskbar to display dialogs or to manipulate the top level windows.

Options:

- Cascade** Cascade all top level windows.
- Computers** Display the Find Computers dialog (requires Active Directory Domain Services)
- Control** Display the Control panel.
- Customize** Display the Customize Taskbar dialog.
- Date** Display the Date and Time dialog.

Desktop	Show the Windows desktop.
Help	Display the Help and Support Center dialog.
HTile	Horizontally tile all top level windows.
Lock	Toggle the taskbar lock.
Min	Minimize all windows.
Max	Maximize all windows.
Printers	Display the Printers and Faxes dialog.
Properties	Display the Taskbar Properties dialog.
Run	Display the Run dialog.
Search	Display the Search dialog.
Shutdown	Display the Shut Down Computer dialog.
Start	Display the Start Menu.
Task	Display the Windows Task Manager dialog.
VTile	Vertically tile all top level windows.

4.3.146 TASKDIALOG

Purpose: Display a Windows Task Dialog

Format: TASKDIALOG [/*I* /*S* /*W*] button*type* "title" "instruction" [*text*]

button<i>type</i>	One or more of OK , YES , NO , RETRY , CANCEL , and/or CLOSE
title	Text for the task dialog title
instruction	Text for the main instruction
text	Optional additional text that appears below the main instruction, in a smaller font

/I(nformation icon)
/S(top icon)

/W(arning icon)

See also: [INKEY](#), [INPUT](#), [MSGBOX](#) and [QUERYBOX](#)

Usage:

TASKDIALOG requires Windows Vista or later.

The button the user chooses is indicated using the internal variable [%_?](#). Be sure to save the return value in another variable or test it immediately; because the value of [%_?](#) changes with every internal command. The following list shows the value returned for each button:

response	% ?
Yes or OK	10
No	11
Cancel or Close	12
Retry	13

If there is an error in the TASKDIALOG command itself, [% ?](#) will be set to 2.

For example, to display a Yes / No message box and take action depending on the result, you could use commands like this:

```
taskdialog yes no "Copy" "Copy all files to A:?"  
if %_? == 10 copy * a:
```

Since TASKDIALOG doesn't write to standard output, it disables redirection allow you to enter the redirection characters (< and >) in your prompt text. If you want to use pipe characters or command separators, you will need to escape or quote them.

TASKDIALOG creates a popup dialog box. If you prefer to retrieve input from the command line, see the [INKEY](#) and [INPUT](#) commands.

Options:

- /I** Display an icon consisting of a lower case "i" in a circle in the message box.
- /S** Display a stop sign icon in the message box.
- /W** Display an exclamation point icon in the message box.

4.3.147 TASKEND

Purpose: End the specified process

Format: `TASKEND [/F] pid | name | "title"`

pid	The process ID
name	The process name
title	Window title

[/F\(orce\)](#)

See also: [TASKLIST](#), [PID](#), [DETACHPID](#), [WINTITLE](#)

Usage:

Windows applications (and Windows itself) run as one or more processes or tasks. You can use the [TASKLIST](#) command to display a list of currently-running tasks. [TASKEND](#) can be used to end a task.

When you use [TASKEND](#), you must specify the task you want to end by process ID number, by name (usually the name of the executable file that started the task) or by window title. If you use the Window title to specify the task, you must enclose it in double quotes. You can use wild cards and extended wildcards in the window title.

If you use TASKEND without the **/F** option, the effect is much the same as closing a window by clicking the close button. The application is notified of the request to end the task and has an opportunity to save data, prompt whether you mean to shut down, and perform other normal "close" operations.

If you use the **/F** option with TASKEND, the application is shut down abruptly and has no chance to save data. Use of the **/F** option is only recommended for unusual circumstance and advanced users because of the possibility of data loss.

Using this command may require the Windows DEBUG privilege, so (depending on the Windows version and the process you are trying to end) it may not work in a limited user account.

Option:

- /F** Forces the task or application to end immediately, with no opportunity to save data, prompt the user, etc. Use this option with caution; it can possibly lead to system instability and data loss or corruption.

4.3.148 TASKLIST

Purpose: Display a list of active processes

Format: TASKLIST [/C /D /L /M /N /O /P /T] [name]

name Process name or window title

/C (Priority)	/N (class names)
/D (show modules)	/O(rder by PID)
/L (Startup command)	/P(ause)
/M(emory)	/T(ime)

See also: [TASKEND](#).

Usage:

Windows programs run as one or more processes or tasks. You can use the TASKLIST command to display a list of currently-running tasks. TASKLIST displays the process ID number for each running task, the name of the executable program that started the task, and, when available, the window title. You can also optionally display the process priority, the modules (dll's) loaded by that process, the startup command line, the memory usage, the class name of the main window of the process, and the cpu usage.

If **name** begins with a =, it is assumed to be a process ID instead of a process name or window title.

TASKLIST will display a * after the process ID of the current process.

You can limit the output of TASKLIST by specifying the task name that you wish to see. The name can contain [wildcards and extended wildcards](#).

Options:

- /=** Display the TASKLIST command dialog to help you set the command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

/C Display the current priority class for each process. The priority values are (in hex):

8000	Above normal
4000	Below normal
100	Realtime
80	High
40	Idle
20	Normal

/D Display the loaded modules for each process.

/L Display the startup command line for each process.

/M Display the memory usage for each process.

/N Display the class name for the main window of each process.

/O Sort the output by Process ID (PID).

/P Wait for a key to be pressed after each screen page before continuing the display.

/T Display the system and user cpu usage for each process.

4.3.149 TCDIALOG

Purpose: Display the command dialogs

Format: TCDIALOG command

command The dialog for the command to execute

See also: [Command Dialogs](#).

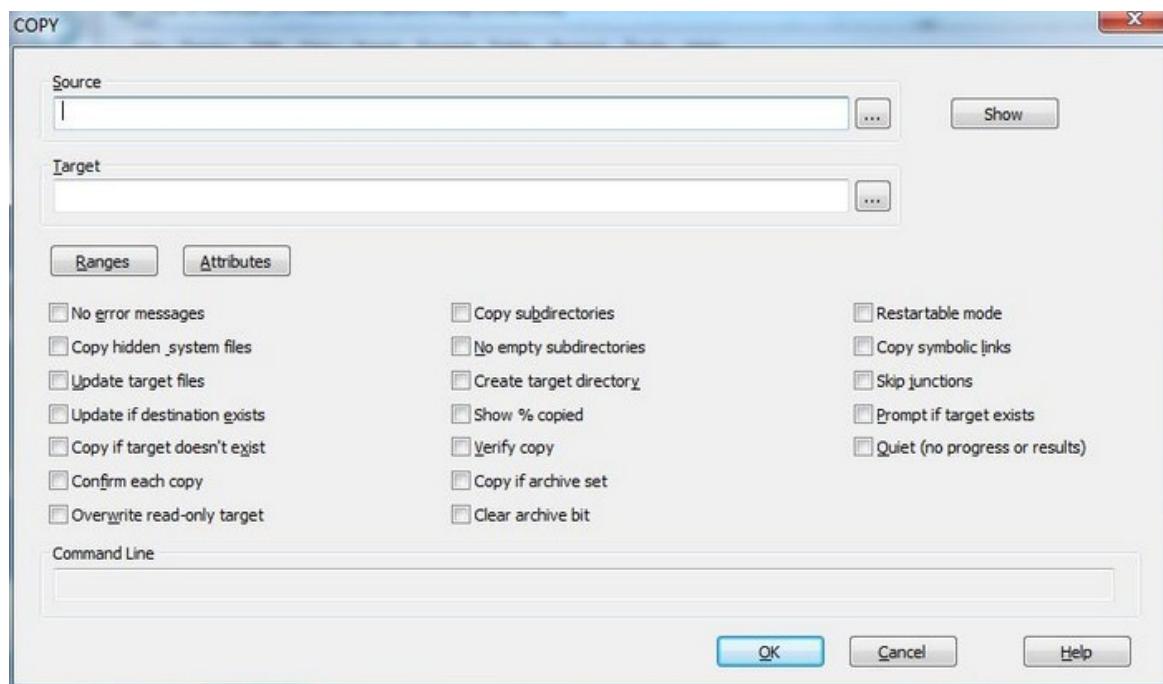
Usage:

Most of the file commands (i.e., COPY, DEL, DIR, MOVE, REN, etc.) have an associated dialog that will interactively build the command line for you, without requiring you to memorize switches and the order of options. You can invoke those dialogs with TCDIALOG, with the **/=** command line option, or with Alt-F1 at the command line.

If **command** does not have an associated dialog, TCDIALOG returns a usage error.

For example, to display the dialog for COPY:

```
tcdialog copy
```



The "Show" button in a command dialog will show all of the files that match the file specification in the edit field to the left. This may include subdirectories if you've selected that option (for example, in COPY or DIR), so it may take a few seconds to populate the list before displaying it.

The internal commands that have a command dialog are:

[ATTRIB](#)
[COPY](#)
[DEL](#)
[DESCRIBE](#)
[DIR](#)
[DIRHISTORY](#)
[EVENTLOG](#)
[GLOBAL](#)
[HEAD](#)
[HISTORY](#)
[IFTP](#)
[JABBER](#)
[LIST](#)
[MD](#)
[MKLINK](#)
[MKLNK](#)
[MOVE](#)
[PLAYAVI](#)
[PLAYSOUND](#)
[PLUGIN](#)
[PRIORITY](#)
[RD](#)
[REN](#)
[SELECT](#)

[SENDHTML](#)
[SENDMAIL](#)
[START](#)
[SYNC](#)
[TAIL](#)
[TAR](#)
[TASKLIST](#)
[TOUCH](#)
[TREE](#)
[TYPE](#)
[UNTAR](#)
[UNZIP](#)
[VIEW](#)
[ZIP](#)

4.3.150 TCFILTER

Purpose: Display or change the Take Command List View filter

Format: TCFILTER [/C] filter

filter A wildcard string or regular expression

[/C\(lear\)](#)

See also: [TCFILTER](#)

Usage:

TCFILTER allows you to set the filter used by the **Take Command** List View to determine what file and folder names to display. For example, to only display files with a .DOC extension in the List View:

tcfilter *.doc

The filter can either use the **TCC** extended [wildcard](#) syntax or (if preceded by ::) a regular expression. See [Regular Expression Syntax](#) for details on valid regular expressions.

Option:

/C Clear the current filter

4.3.151 TCTOOLBAR

Purpose: Change the Take Command tool bar buttons

Format: TCTOOLBAR [/C /I /R filename /U /W filename] tab button, flags, icon, label, title, directory, command

tab	The label of the toolbar tab
button	The button number (1 - 50)
flags	0=Start new tab, 1=Send to current tab, 2=Change Folders directory
icon	Icon to display on button
label	The button label
title	The tab title (when starting new tabs)

directory The startup or Folders directory
command The command to execute or keystrokes to send

<u>/C(lear)</u>	<u>/U(pdate)</u>
<u>/I (reset toolbar)</u>	<u>/W (save to file)</u>
<u>/R(ead file)</u>	

Usage:

TCTOOLBAR lets you configure the **Take Command** tab tool bar buttons (you can also use the [Configure Tool Bar dialog](#) available from the [Options](#) menu). The changes you make can be temporary or, with the **/U** option, written to the **TCMD.INI** file so that they will be loaded the next time **Take Command** starts.

There are a maximum of 50 buttons on the tab tool bar. The **button** parameter must be a number from 1 to 50 to select the button you want to work with. If you enter a command like

```
tctoolbar 1
```

the button with that number will be removed from the tool bar. If you want to add or modify a button, you must include the **flags**, **icon** and/or **label**, and **command** parameters.

The **flags** parameter specifies what happens when you click the button. If **flags** is 256, **Take Command** will use **command** to start a new tab (or a new window if **command** is a GUI app). If **flags** is 257, the **command** text (in [KEYSTACK](#) format) is sent to the current tab. If **flags** is 258, the button will change the default directory in the **Folders** view. You can optionally add 4 to the value of **flags** to insert a separator before the button.

The **icon** parameter allows you to specify the name of an icon file (or an executable file if you want to use its default icon). The icon will be displayed to the left of the button label. If you have entered a **label**, the **icon** parameter is optional.

The **label** parameter specifies the text that appears on the button. If the text contains white space or other special characters, it must be enclosed in double quotes. If you have entered an **icon** file, **label** is optional.

The optional **title** parameter specifies the new tab title (if **flags=0**).

If you're starting a new window, the **directory** parameter will set the startup directory for the command. If you are changing the **Folders** directory, the **directory** parameter specifies the new directory.

The **command** parameter contains the command to start a new tab (if **flags=0**), or the keystrokes to be sent to the current tab in [KEYSTACK](#) format (if **flags=1**) when the button is clicked.

Option:

- /C** Clear all entries from the toolbar.
- /I** Reset the toolbar to the definition in TCMD.INI. (Not available in **TCC/LE**.)
- /R** Load the toolbar button definitions from the specified file. **/R** will **not** clear an existing toolbar; you must use **/C** for that. The file should be in the same format as the **[Toolbars]** section in TCMD.INI:

```
[Toolbar1]
Title=MyTabs
Bn=flags,icon,label,title,directory,command
```

n - the button number (1 - 50)
flags - 256=start new tab (or new window if a GUI app), 257=send keystrokes to current tab, 258=Change Folders directory
icon - the icon to display on the label (leave empty for no icon)
label - the label to display on the button
title - the tab title (if starting a new tab)
directory - startup directory or Folders directory
command - the command to execute

The **command** and **directory** parameters can include environment variables, internal variables, and variable functions. Note that the variable expansion occurs in **Take Command**, not **TCC**, so internal variables like [%_cwd](#) will not probably work as expected.

- /U Write the changed button definition to the **TCMD.INI** file so that it will be included the next time **Take Command** starts.
- /W Save the current toolbar to the specified file. (Not available in **TCC/LE**.)

4.3.152 TEE

Purpose: Copy standard input to both standard output and a file

Format: TEE [/A /D /T] file...

file One or more files that will receive the "tee-d" output.

/A(ppend)	/T(ime)
/D(ate)	

See also: [Y](#), [piping](#) and [redirection](#).

Usage:

TEE is normally used to "split" the output of a program so that you can see it on the display and also save it in a file. It can also be used to capture intermediate output before the data is altered by another program or command.

TEE gets its input from standard input (usually the piped output of another command or program), and sends out two copies: one to standard output, the other to the **file(s)** that you specify. TEE is not likely to be useful with programs which do not use standard output, because these programs cannot send output through a pipe.

For example, to search the file **DOC** for any lines containing the string **Take Command**, make a copy of the matching lines in **TC.DAT**, sort the lines, and write them to the output file **TCS.DAT**:

```
ffind /t"Take Command" doc | tee tc.dat | sort > tcs.dat
```

If you are typing at the keyboard to produce the input for TEE, you must enter a **Ctrl-Z** to terminate the input.

See [Piping](#) for more information on pipes.

Option:

- /A Append to the file(s) rather than overwriting them.
- /D Prefix each line with the current date (in yyyy-mm-dd format).
- /T Prefix each line with the current time (in hh:mm:ss.ms format).

4.3.153 TEXT

Purpose: Display a block of text in a batch file

Format: TEXT

.

ENDTEXT

See also: [ECHO](#), [ECHOS](#), [SCREEN](#), [SCRPUT](#), and [VSCRPUT](#).

Usage:

TEXT can only be used in batch files. Both TEXT and ENDTEXT must be entered as the only commands on their respective lines, and cannot be included in a [command group](#).

The TEXT command is useful for displaying menus, tables, special characters, or multiline messages. TEXT will display all lines in the batch file between itself and the terminating ENDTEXT. The display starts at the current display position, which allows you to start its display with other text, e.g., from the [ECHOS](#) command.

The lines between TEXT and ENDTEXT are not parsed. As a consequence, no environment variable expansion or other processing is performed, and all lines are displayed exactly as they are stored in the batch file, subject only to the choice of font and codepage differences, if any, between the program which created the file and that in effect during its execution. This makes it easy to include special characters, e.g., < | > in the text. However, if the ANSI X3.64 interpretation option is enabled, you can change screen colors by inserting ANSI X3.64 escape sequences anywhere in the text block. The ENDTEXT command itself will not be displayed.

You can also use the [CLS](#) or the [COLOR](#) command to set the default screen colors before executing TEXT.

Redirecting TEXT output

To redirect or pipe the entire block of text, use [redirection](#) or [piping](#) on the TEXT command itself as shown in the [Examples](#) below. As with any other command, this redirection is not affected by redirection of all output of the batch file by the command which started the batch file. Attempting to redirect or pipe the actual text lines is ignored. Attempting to redirect or pipe the ENDTEXT line is invalid.

Warning: If the TEXT command is redirected or piped, and the redirection/piping fails, the lines of the batch file following the TEXT command are executed as if they were commands, causing potential harm.

The simplest way to avoid trouble this may cause is to use the [ON ERROR](#) command before TEXT. See the second example below.

Examples

The following batch file fragment displays a simple menu:

```
@echo off & cls
screen 2 0
text
Enter one of the following:
 1 - Spreadsheet
 2 - Word Processing
 3 - Utilities
 4 - Exit
endtext
inkey /k"1234" Enter your selection: %%key
```

The example below uses TEXT to display or append to a file (specified as the optional parameter of the batch file):

```
@echo off
setlocal
setdos /x-6
set dest=%@if[%# GT 0,>> %1,]
setdos /x+6
set repeat=0
on error (unset dest & goto PROBLEM)
:PROBLEM
iff %repeat GT 1 then
  echo Repeated problems - quitting
  quit
endiff
set repeat=%@inc[%repeat]
text %dest
+-----+
| Logical Drives |
+-----+
endtext
subst %dest
echo. %dest
if %_transient eq 1 .and. %# EQ 0 pause
endlocal
```

4.3.154 TIME

Purpose: Display or set the system time

Format: TIME [/S [server] /T /U] [hh[:mm:ss]]] [AM | PM]

hh The hour (0 - 23)
mm The minute (0 - 59)

ss The second (0 - 59)

[/S\(erver time\)](#)
[/T \(Display only\)](#)

[/U \(UTC time\)](#)

See also: [DATE](#).

Usage:

If you don't enter any parameters, TIME will display the current system time and prompt you for a new time. Press Enter if you don't wish to change the time; otherwise, enter the new time:

```
[c:\] time
Thu Aug 18, 2011 9:30:06
Enter new date (mm-dd-yy) :
```

TIME defaults to 24-hour format, but you can optionally enter the time in 12-hour format by appending **a**, **am**, **p**, or **pm** to the time you enter. For example, to enter the time as 9:30 am:

```
time 9:30 am
```

Options:

- /S server** Sets the date and time from the specified internet time server. If no server is specified, TIME uses the server defined in the [Time Server](#) configuration option (the default is `clock.psu.edu`).
- /T** Displays the current time but does not prompt you for a new time. You cannot specify a new time on the command line with **/T**. If you do, the new time will be ignored.
- /U** Display or enter the UTC time. (Not available in **TCC/LE**.)

4.3.155 TIMER

Purpose: TIMER is a system stopwatch

Format: TIMER [/1 /2 /3 /Q /S] [ON | OFF] [command]

ON	Force the stopwatch to reset and start
OFF	Force the stopwatch to stop
command	Time the specified command

/1	stopwatch #1 (default)	/Q	quiet
/2	stopwatch #2	/S	split
/3	stopwatch #3		

Usage:

The TIMER command accepts its parameters in any order, and acts on the specified one of three possible timers (system stopwatches) by turning it on or off, or by displaying its current elapsed time. The TIMER command with neither of the keywords **ON** and **OFF** nor the **/S** option toggles the state of the timer.

The switch arguments (/1, /2, /3, /Q, and /S) must appear before any other arguments on the TIMER command line.

If you execute TIMER or TIMER /S when the timer is off, or execute TIMER ON at any time, the current time of day is displayed, and the stopwatch starts from :

```
[c:\] timer  
Timer 1 on: 12:21:46
```

If you execute TIMER /S when the timer is on, the elapsed time is displayed:

```
[c:\] timer /s  
Timer 1 Elapsed time: 0:00:12.06
```

If you execute TIMER when it is on, or execute TIMER OFF, the stopwatch stops, the current time and the elapsed time are displayed, and the elapsed time is reset:

```
[c:\] timer  
Timer 1 off: 12:21:58  
Elapsed time: 0:00:12.06
```

There are three stopwatches available (1, 2, and 3) so you can time multiple overlapping events. By default, TIMER uses stopwatch #1.

TIMER is particularly useful for timing events in batch files. For example, to time both an entire batch file, and an intermediate section of the same file, you could use commands like this:

```
rem Turn on timer 1  
timer  
rem Do some work here  
rem Turn timer 2 on to time the next section  
timer /2  
rem Do some more work  
echo Intermediate section completed  
rem Display time taken in intermediate section  
timer /2  
rem Do some more work  
rem Now display the total time  
timer
```

You can optionally specify a command for TIMER to run. This is the equivalent of "timer on & command & timer off". For example:

```
timer dir c:\ /s
```

The smallest interval TIMER can measure depends on the operating system you are using, your hardware, and the interaction between the two. However, it should never be more than 60 ms.

You can also retrieve the elapsed time of a timer using the [@TIMER\(\)](#) function.

Options:

- /1 Use timer #1 (the default).
- /2 Use timer #2.
- /3 Use timer #3.
- /Q Don't display any messages.
- /S Display a split time without stopping the timer. To display the current elapsed time but leave the timer running:

```
[c:\] timer /s  
Timer 1 elapsed: 0:06:40.63
```

- ON** Start the timer regardless of its previous state (on or off). Otherwise the TIMER command toggles the timer state (unless **/S** is used).
- OFF** Stops the timer.

4.3.156 TITLE

Purpose: Change the window title

Format: TITLE [/P] title

[/P\(rompt characters\)](#)

title The new window title.

See also: the [TITLEPROMPT](#) variable and the [ACTIVATE](#) and [WINDOW](#) commands.

Usage:

TITLE changes the text that appears in the caption bar at the top of the **TCC** window. You can also change the window title with the WINDOW command or the ACTIVATE command.

The title text should not be enclosed in quotes unless you want the quotes to appear as part of the actual title.

To change the title of the current window to "Title Test":

```
title Title Test
```

Options:

/P Support the special characters in [PROMPT](#).

4.3.157 TOUCH

Purpose: Change a file's [time stamps](#), and optionally create a file

Format: TOUCH [/A:[[-][+]]rhsdaecjot] /C [/D[acw][date]] /E /F /I""text"" /N /O:[-]adegnrstu /Q /R[:acw] file /S[[+]n] /T[acw[u]][hh:mm:ss[.dd]]] file...

file One or more files whose date and/or time stamps are to be changed.

<u>/A:</u>	Attribute select	<u>/N</u>	No action
<u>/C</u>	Create file	<u>/O</u> ...	Order
<u>/D</u>	Date	<u>/Q</u>	Quiet
<u>/E</u>	No error messages	<u>/R</u>	Reference file
<u>/F</u>	Force read-only files	<u>/S</u>	Subdirectories
<u>/I</u>	Match descriptions	<u>/T</u>	Time

File Selection:

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), subdirectories, [catalog files](#), and [include lists](#).

Usage:

TOUCH is used to change the date and / or time of a file. You can use it to be sure that particular files are included or excluded from an internal command, backup program, compiler MAKE utility, or other program that selects files based on their time and date stamps, or to set a group of files to the same date and time for consistency. If you don't specify any arguments, TOUCH will display its command dialog.

TOUCH should be used with caution, and in most cases should only be used on files you create. Many programs depend on file dates and times to perform their work properly. In addition, many software manufacturers use file dates and times to signify version numbers. Indiscriminate changes to date and time stamps can lead to confusion or incorrect behavior of other software.

By default, TOUCH affects only files. You must utilize the /A: option to include directories. /A:D will select directories only.

TOUCH sets three internal variables:

<u>%_touch_dirs</u>	The number of directories touched
<u>%_touch_files</u>	The number of files touched
<u>%_touch_errors</u>	The number of errors

Options:

/= Display the TOUCH command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

/A: Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:.

You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

/C Create **file** (as a zero-byte file) if it does not already exist. You cannot use wildcards with **C**, but you can create multiple **files** by listing them individually on the command line.

/D If neither /R nor /D are specified, the current date is used. If the /D option is specified without

date, TOUCH will not modify the date even if [/R](#) is also specified. If the [/D](#) option is followed by date, and [/R](#) is not specified, date is used. The date must not be quoted. If both [/R](#) and [/D](#) with date are specified, the one specified later in the command takes effect.

On an LFN drive, you can specify which of the date fields should be set by appending **a**, **c**, or **w** to the [/D](#) option:

- a** Last access date
- c** Creation date
- w** Last modification (write) date

If you append a **u** to the date field, TOUCH will set the UTC date rather than the local date.

- /E** Suppress all non-fatal error messages, such as "File not found." Fatal error messages, such as "Drive not ready," will still be displayed. This option is most useful in batch files.
- /F** The file systems normally do not permit changing timestamps of read only files. The [/E](#) option forces date and time change of read-only files by temporarily removing the read only attribute.
- /I"text"** Select files by matching text in their descriptions. See [Description Ranges](#) for details.
- /N** Display what would occur without actually doing it.
- /O:...** Sort the files before processing. You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:
 - n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
 - Reverse the sort order for the next sort key
 - a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
 - d** Sort by date and time (oldest first); also see [/T:acw](#)
 - e** Sort by extension
 - g** Group subdirectories first, then files
 - r** Reverse the sort order for all options
 - s** Sort by size
 - t** Same as **d**
 - u** Unsorted
- /Q** Do not display normal messages.
- /R** The [/R](#) option permits duplication of the time stamp of **ref_file** (which must immediately follow the [/R](#), and can be a file or subdirectory). For example, if you recompile an old program (e.g., to obtain an intermediate file that has long been deleted) you may want to use the timestamp of the source file that was last changed as the time stamp of the newly built duplicate of the original object file to prevent a "make" from attempting to rebuild everything else in the project as shown in the example:

```
touch /r project.c project.obj
```

Another use could be to synchronize files without rendering the current version inaccessible

during the synchronization:

```
touch /c /r c:\jpsoft\tcmd.chm %temp\tcmd.chm
copy /u ftp://ftp.jpsoft.com/help/tcmd.chm %temp\tcmd.chm
```

In the above example TOUCH creates an empty file with the time stamp of your already existing help file; [COPY](#) updates the empty file if a newer version is available (beware of time stamp synchronization across the Internet!).

On an LFN drive, you can specify which of the date/time fields should be used by appending **a**, **c**, or **w** to the **/R** option:

- a** Last access date and time (on VFAT volumes access time is always midnight).
- c** Creation date and time
- w** Last modification (write) date and time

/S TOUCH all matching files in the specified directory and its subdirectories. Do not use **/S** with @file lists. See [@file lists](#) for details.

If you specify a number after the **/S**, TOUCH will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\l\e", **/S2** will only affect the "a", "b", and "c" directories.

If you specify **a +** followed by a number after the **/S**, TOUCH will not modify any time stamps until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\l\e, **/S+2** will not modify anything in \a or \a\b. (Not available in **TCC/LE**.)

/T If neither [/R](#) nor [/I](#) are specified, the current time is used. If the [/I](#) option is specified without time, TOUCH will not modify the time even if [/R](#) is also specified. If the [/I](#) option is followed by time, and [/R](#) is not specified, time is used. (Time must not be quoted). If both [/R](#) and [/I](#) with time are specified, the one specified later in the command takes effect.

On an LFN drive, you can specify which of the time fields should be set by appending **a**, **c**, or **w** to the **/T** option:

- a** Last access time (on VFAT volumes access time is always midnight).
- c** Creation time
- w** Last modification (write) time

If you append a **u** to the time field, TOUCH will set the UTC time rather than the local time.

4.3.158 TPIPE

Purpose: Text filtering, search, substitution, conversion, and sorting

Format: See Options below.

Usage:

TPIPE does text filtering, substitution, conversion, and sorting on text files. If you don't specify an input filename, TPIPE will read from standard input if it has been redirected. If you don't specify an output filename, TPIPE will write to standard output. This is substantially slower than reading from and writing to

files, but allows you to use TPIPE with pipes.

You can specify multiple filters, which will be processed in the order they appear on the command line. Do not insert any unquoted whitespace or switch characters in the arguments to an option.

Row and column positions start at 1.

If you need to process a Windows Unicode UTF-16 file, unless the filter supports Unicode directly (for example, **/simple**) you'll need to convert it to UTF-8 first (see **/unicode=...**).

Options:

/input=filename

Filename to read. This can be either a disk file, file list (@filename), or CLIP:. If it is not specified, TPIPE will read from standard input.

/output=filename

Filename to write. This can be either a disk file or CLIP:. If it is not specified, TPIPE will write to standard output.

/outputfolder=directory

Set the output filter directory.

/inputbinary=n

Determines how binary files are processed. The options are:

0 - Binary files are processed (default)

1 - Binary files are skipped

2 - Binary files are confirmed before processing

/inputdelete=n

If 1, the input files will be deleted after processing. **USE WITH CAUTION!**

/inputprompt=n

If 1, TPIPE will prompt before processing each input files.

/outputchanged=n

Sets the output changed mode. The options are:

0 - Always output

1 - Only output modified files

2 - Delete original if modified

/outputmode=n

If **n=1**, TPIPE will open each output file in its associated program upon completion.

0 - Output to clipboard (all files are merged)

- 1 - Output to files
- 2 - Output to a single merged file

/outputopen=n

Sets the output mode. The options are:

/clipboard

Runs the current filter with input from and output to the clipboard.

/filter=filename

Name of filter file to load (see /save=filename)

/save=filename

Saves the filter settings defined on the command line to the specified filename, and returns without executing any filters.

/startsubfilters

The following filters are created as sub filters, until the closing /ENDSUBFILTERS. Sub filters allow a restricted part of the entire text to be operated on by a group of filters without effecting the entire text. For example, a "Restrict to delimited fields" (CSV, Tab, Pipe, etc.) filter can pick out a range of CSV fields, and then a search/replace filter can operate JUST on the text restricted.

/endsubfilters

End the sub filters defined by the preceding /STARTSUBFILTERS.

/buffersize=n

Sets the buffer size for the preceding search/replace filter. (The default is 4096.)

/editdistance=n

Sets the edit distance threshold for the preceding search/replace filter. (The default is 2.)

/comment=text

Add a comment to a filter file.

Text - Comment to add

/
database=Mode,GenerateHeader,Timeout,Connection,InsertTable,FieldDelimiter,Qualifier

Adds a database-type filter.

Mode

0 Delimited output

- 1 Fixed width
- 2 XML
- 3 Insert script

GenerateHeader - Generates header information when True.

Timeout - SQL command timeout in seconds.

ConnectionStr - The database connection string.

InsertTable - The name of the insert table.

FieldDelimiter - The string to use between columns.

Qualifier - The string to use around string column values.

/dup=Type,MatchCase,StartColumn,Length,IncludeOne,Format

Remove or show duplicate lines. The arguments are:

Type:

- 0 - Remove duplicate lines
- 1 - Show duplicate lines

MatchCase - If 1, do case-sensitive comparisons

StartColumn - The starting column for comparisons

Length - The Length of the comparison

IncludeOne - Include lines with a count of 1

Format - how the output should be formatted for Type=1. For example, "%d %s" to show the count followed by the string.

/eol=input,output,length

Add an EOL (end of line) conversion filter. The arguments are:

input:

- 0 - Unix (LF)
- 1 - Mac (CR)
- 2 - Windows (CR/LF)
- 3 - Auto

If you are unsure of the source, select Auto. The Auto option can detect and modify text files containing a variety of line endings.

4 - Fixed (use the length parameter to specify the length)

If you are converting a mainframe file that contains fixed length records, select "Fixed length" and enter the record length. The maximum record length is 2,147,483,647 characters. Note: If you are converting 132 column mainframe reports, you should set the fixed length to 133, because each line has a prefix character.

output:

- 0 - Unix
- 1 - Mac
- 2 - Windows
- 3 - None

length - The line length to use if input=4

/file=type,MatchCase,filename

Add a file-type filter. The arguments are:

type:

- 17 Restrict to filenames matching the Perl pattern
- 18 Restrict to filenames NOT matching the Perl pattern

MatchCase - If 1, do a case sensitive match (where appropriate)

filename - the filename to use

/grep=Type,IncludeLineNumbers,IncludeFilename,MatchCase,CountMatches,PatternType,UTF8,IgnoreEmpty,Pattern

Adds a Grep type line based filter. The arguments are:

Type:

- 0 Restrict lines matching (for subfilters)
- 1 Restrict lines NOT matching (for subfilters)
- 2 Extract matches
- 3 Extract matching lines (grep)
- 4 Extract non-matching lines (inverse grep)
- 5 Remove matching lines
- 6 Remove non-matching lines

IncludeLineNumbers - 1 to include the line number where the pattern was found

IncludeFilename - 1 to include the filename where the pattern was found

MatchCase - 1 to do a case-sensitive comparison when matching the pattern

CountMatches - 1 to output a count of the number of matches

PatternType

- 0 Perl pattern
- 1 Egrep pattern
- 2 Brief pattern
- 3 MS Word pattern

UTF8 - 1 to allow matching Unicode UTF8 characters

IgnoreEmpty - 1 to ignore empty matches

Pattern - the (regular expression) pattern to match

/head=Exclude,LinesOrBytes,Count

Add a head type filter (includes or excludes text at the beginning of the file). The arguments are:

Exclude:

- 0 - Include the text
- 1 - Exclude the text

LinesOrBytes:

- 0 - Measure in lines
- 1 - Measure in bytes

Count - the number of lines or bytes to include or exclude

/insert=type,position,string

Add an insert type filter. The arguments are:

type:

- 0 - Insert column

Inserts a new column of text. The position the text is inserted is determined by a column count. The leftmost column is column 1 – inserting in this column displaces all other text to the right. If the insert column given is 0, the text is inserted at the end of the line. If the insert column is negative, the text is inserted at the given position relative to the end of the line. If the insert column given is before the start of the line, or beyond the end of the line, then the text is prepended or appended to the line respectively. Note - this filter is designed for ANSI or Unicode UTF-8 data - it will not handle UTF-16 data. If you need to process UTF-16 files, convert them to UTF-8 first and then convert back to UTF-8 after doing the insertion.

- 1 - Insert bytes

Insert bytes at the given offset (from 0 to the size of the file).

position - the position to insert the string

string - the string to insert

/line=LineNumber,Increment,SkipBlank,DontNumberBlank,NumberFormat[,DontReset]

Adds a Line Number filter. The arguments are:

LineNumber - the starting line number

Increment - the amount to add for each new line number

SkipBlankIncrement - don't increase the line number for blank lines

DontNumberBlank - don't put a line number on blank lines

NumberFormat - The format to use for the line number. The format syntax is:

[-.][width][.precision]d

An optional left justification indicator, ["-"]

An optional width specifier, [width] (an integer). If the width of the number is less than the width specifier, it will be padded with spaces.

An optional precision specifier [precision] (an integer). If the width of the number is less than the precision, it will be left padded with 0's.

The conversion type character:

d - decimal

DontReset - if 1, do not reset the line count at the end of the file. The default is 0.

/log=Filename

Log the TPIPE actions.

Filename - Name of log file

/maths=operation,operand

Adds a maths type filter.

operation - the operation to perform

- | | |
|----|------------------------------------|
| 0 | + |
| 1 | - |
| 2 | * |
| 3 | div (the remainder is ignored) |
| 4 | mod (the remainder after division) |
| 5 | xor |
| 6 | and |
| 7 | or |
| 8 | not |
| 9 | shift left (0 inserted) |
| 10 | shift right (0 inserted) |
| 11 | rotate left |
| 12 | rotate right |

operand - the operand to use

/merge=type,filename

Adds a merge type filter (merge into single output filename). The arguments are:

type:

- 0 Merge into filename
- 1 Retain lines found in filename
- 2 Remove lines found in filename
- 3 Link filter filename

filename - the filename to use

/number=type,value

Add a number-type filter. The arguments are:

type:

- 0 Convert Tabs to Spaces
- 1 Convert Spaces to Tabs
- 2 Word wrap (value column width)
- 3 Pad to width of value
- 4 Center in width of value
- 5 Right justify in width of value
- 6 Restrict CSV field to value
- 7 Restrict tab-delimited field to value
- 8 Truncate to width value
- 9 Force to width value
- 10 Repeat file value times
- 11 Restrict to blocks of length
- 12 Expand packed decimal (with implied decimals)
- 13 Expand zoned decimal (with implied decimals)
- 14 Expand unsigned (even-length) packed decimal
- 15 Expand unsigned (odd-length) packed decimal

Value - the numeric value to use

/perl=BufferSize,Greedy,AllowComments,DotMatchesNewLines

Sets the Perl matching options for the immediately preceding search/replace filter.

BufferSize - The maximum buffer size to use for matches. Any match must fit into this buffer, so if you want to match larger pieces of text, increase the size of this buffer to suit. Default is 4096.

Greedy - If the pattern finds the longest match (greedy) or the shortest match. Default is false.

AllowComments - Allow comments in the Perl pattern. Default is false.

DotMatchesNewLines - Allow the '.' operator to match all characters, including new lines. Default is true.

/

replace=Type,MatchCase,WholeWord,CASEReplace,PromptOnReplace,Extract,First Only,SkipPromptIdentical,Action,SearchStr,ReplaceStr

Adds a search and replace (find and replace) filter. Search / Replace lists discard blank search terms and terms where the replacement is identical to the search. Search / Replace lists can generate log entries (useful for debugging). Logs can optionally be output only for where replacements occurred.

The arguments are:

Type:

- 0 Replace
- 1 Pattern (old style)
- 2 Sounds like
- 3 Edit distance

- 4 Perl pattern
- 5 Brief pattern
- 6 Word pattern

MatchCase - Matches case when set to 1, ignores case when set to 0

WholeWord - Matches whole words only when set to 1

CaseReplace - Replaces with matching case when set to 1

PromptOnReplace - Prompts before replacing when set to 1

Extract - If 1, all non-matching text is discarded

FirstOnly - If 1, only replace the first occurrence

SkipPromptIdentical - If 1, don't bother prompting if the replacement text is identical to the original.

Action - the action to perform when found:

- 0 replace
- 1 remove
- 2 send to subfilter
- 3 send non-matching to subfilter
- 4 send subpattern 1 to subfilter etc

SearchStr - the string to search for

ReplaceStr - the string to replace it with

```
/  
replacelist=Type,MatchCase,WholeWord,CaseReplace,PromptOnReplace,FirstOnly,Skip  
PromptIdentical,Simultaneous,LongestFirst,Filename
```

Add a search and replace list, using search and replace pairs from the specified file.

Type:

- 0 Replace
- 1 Pattern (old style)
- 2 Sounds like
- 3 Edit distance
- 4 Perl pattern
- 5 Brief pattern
- 6 Word pattern

MatchCase - Matches case when set to 1, ignores case when set to 0

WholeWord - Matches whole words only when set to 1

CaseReplace - Replaces with matching case when set to 1

PromptOnReplace - Prompts before replacing when set to 1

FirstOnly - If 1, only replace the first occurrence

SkipPromptIdentical - If 1, don't bother prompting if the replacement text is identical to the original.

Simultaneous - If 1, all search strings are scanned for simultaneously instead of consecutively. (This is useful if the search strings and results strings overlap.)

LongestFirst - If 1, searches for long phrases (most specific) before short phrases (least specific) - this is generally used for translations.

Filename - The file to load search/replace pairs from. If the file extension is .XLS or .XLSX, the file is assumed to be Excel format, if the extension is .TAB the file is assumed to have tab-delimited values, and any other extension (including .CSV) is assumed to have Comma-Separated Values. The filename can contain environment variables enclosed in % signs e.g. %TEMP%\myfile.txt. TPIPE corrects any doubled backslashes.

/run=InputFileName,OutputFileName,"CommandLine"

Adds a Run External Program filter. The arguments are:

InputFilename - the filename that TextPipe should read from after the External Program writes to it.

OutputFilename - the filename that TextPipe should write to for the External Program to read in.

CommandLine - the command line of the program to run. Should include double quotes around the entire command line.

/script=language,timeout,code

Adds an ActiveX script filter.

language: The language of the script

timeout: The command timeout in seconds

script: The code

/

selection=Type,Locate,Param1,Param2,MoveTo,nDelimiter,CustomDelimiter,HasHeader [,ProcessIndividually]

Type - The type of filter to add:

0 – Remove column:

This filter is used to remove columns of text, given a column specification that describes the position of the column relative to the start or end of the line, and the width of the column. There are several ways to specify the columns (Locate,Param1,Param2) to remove:

0 - Start column, End column. This removes all text including and between the specified columns. Useful for removing column in fixed width data files.

1 - Start column, width. Removes Width characters starting from (and including) column Start.

2 - End column, width. Removes Width characters backwards starting from (and including) column End.

3 - Start column to end of line. Removes all characters from the Start column to the very end of the line. Useful for making a file a uniform width.

4 - Width to end of line. Removes Width characters backwards starting from (and including) the last column.

Note - if you are removing more than one column range, it is easiest to remove ranges from right-to-left so that the position of the columns doesn't change.

1 - Restrict lines (restriction filters require sub filters to have any effect)

2 - Restrict columns (restriction filters require sub filters to have any effect)

3 - Restrict to bytes (restriction filters require sub filters to have any effect)

4 - Restrict to delimited fields (CSV, Tab, Pipe, etc.)

6 - Remove lines:

This filter removes a range of lines. There are several ways to specify the lines (Locate,Param1,Param2) to remove:

0 - Start line, End line. This removes all lines including and between the specified lines.

1 - Start line, width. Removes Width lines starting from (and including) line Start.

2 - End line, width. Removes Width lines backwards starting from (and including) line End.

3 - Start line to end of line. Removes all lines from the Start line to the very end of the line.

4 - Width to end of line. Removes Width lines backwards starting from (and including) the last line.

7 - Remove delimited fields (CSV, Tab, Pipe, etc.):

This filter is used to remove fields delimited by a given character. You can choose a predefined delimiter character (Delimiter), or select your own (CustomDelimiter). The trailing delimiter (if any) is also removed. When Comma (.csv) is chosen, TPIPE automatically handles single and double quoted strings, with embedded line feeds.

First Row Contains Field Names

If the first line of the file contains Field Names, set HasHeader to 1 so that TPIPE can count how many fields are expected. It can then determine if a field has embedded CR/LF characters and spans multiple lines. TPIPE can also determine this without a header if the fields are properly double-quoted - TPIPE will notice the missing double quote and continue reading the record from the following line.

Remove Fields

There are several ways to specify the fields (Locate,Param1,Param2) to remove:

0 - Start field, end field. This removes all text including and between the specified fields.

1 - Start field, width. Removes Width fields starting from (and including) field Start.

2 - End field, width. Removes Width fields backwards starting from (and including) field End.

3 - Start field to end of line. Removes all fields from the Start field to the very end of the line.

4 - Width to end of line. Removes Width fields backwards starting from (and including) the last field.

Note - if you are removing more than one field range, it is easiest to remove ranges from right-to-left so that the position of the fields doesn't change.

9 – Move columns:

TPIPE will move columns to a new position on the line. The new position (MoveTo) is specified assuming that the moved columns have been removed from the line.

10 – Move delimited fields (CSV, Tab, Pipe, etc.):

TPIPE will move CSV-delimited fields to a new position on the line. The new position (MoveTo) is specified assuming that the moved fields have been removed from the line. TPIPE ensures that all the delimiters on the line are correctly maintained, both at the end of the line and where the moved fields are inserted.

Note - this filter is designed for ANSI or Unicode UTF-8 data - it will not handle UTF-16 data. You will need to convert UTF-16 files to UTF-8 first, do the selection, and then convert back to UTF-16.

12 – Copy columns:

TPIPE will copy columns to a new position (MoveTo) on the line. Note - this filter is designed for ANSI or Unicode UTF-8 data - it will not handle UTF-16 data. You will need to convert UTF-16 files to UTF-8 first, do the selection, and then convert back to UTF-16.

13 – Copy delimited fields (CSV, Tab, Pipe, etc.):

TPIPE will copy CSV-delimited fields to a new position (MoveTo) on the line.

TPIPE ensures that all the delimiters on the line are correctly maintained, both at the end of the line and where the copied fields are inserted. Note - this filter is designed for ANSI or Unicode UTF-8 data - it will not handle UTF-16 data. You will need to convert UTF-16 files to UTF-8 first, do the selection, and then convert back to UTF-16.

17 – Remove byte range:

This filter is used to remove a range of bytes. There are several different ways to specify the bytes (Locate,Param1,Param2) to remove:

0 - Start byte, end byte. This removes all text including and between the specified byte.

1 - Start byte, width. Removes Width byte starting from (and including) the start byte.

2 - End byte, width. Removes Width fields backwards starting from (and including) byte End.

3 - Start byte to end of file. Removes all fields from the Start byte to the very end of the file.

4 - Width to end of file. Removes Width fields backwards starting from (and including) the last byte.

Note - if you are removing more than one byte range, it is easiest to remove ranges from right-to-left so that the position of the bytes doesn't change.

Locate - How to determine which areas to affect:

0 - Restrict %d .. %d

1 - Restrict %1:d starting at %0:d

2 - Restrict %1:d starting at END - %0:d

3 - Restrict %d .. END - %d

4 - Restrict END - %d .. END - %d

Param1, Param2 - The integer values for the Locate method.**MoveTo** - The integer value where to move or copy the columns or fields to (first columns or field is 1).**Delimiter** - The index of the standard delimiter to use:

0 - Comma

- 1 - Tab
- 2 - Semicolon
- 3 - Pipe (|)
- 4 - Space
- 5 - Custom

CustomDelimiter - The custom delimiter to use (if Delimiter == 5). This should be a quoted string; if you are not using a custom delimiter then set this field to "".

HasHeader - 1 if the file's first row is a header row, 0 if not.

ProcessIndividually - Whether to apply sub filters to each CSV or Tab field individually (1), or to the fields as one string value (0). The default is false.

/simple=n[u]

Adds a simple filter type. n is the type of filter to add, and for those filters that support it, u indicates that the filter will be dealing with Unicode data.

1 – Convert ASCII to EBCDIC

EBCDIC is the character collating sequence commonly used on mainframes.

Some characters cannot be converted because they exist in one character set but not the other.

2 – Convert EBCDIC to ASCII

3 – Convert ANSI to OEM

Converts from ANSI to ASCII/OEM. ANSI is an 8-bit character set used by Windows, and it includes all accentuated Roman characters used by non-English languages like French, German and Spanish. (Windows uses UTF-16LE for all of its internal APIs, and converts to ANSI if the user is using raster fonts or ANSI files.) ASCII/OEM is an extension of the original IBM character set where various non-essential characters are replaced by language-specific accentuated characters. Different ASCII/OEM character sets are not compatible. They must be converted to ANSI and then back to the correct ASCII/OEM character set to be readable.

4 – Convert OEM to ANSI

5 – Convert to UPPERCASE

Forces all text to UPPERCASE. To make the conversion, the function uses the current language selected by the user in the system Control Panel. If no language has been selected, TPIPE uses the Windows internal default mapping.

6 – Convert to lowercase

Forces all text to lowercase. To make the conversion, the function uses the current language selected by the user in the system Control Panel. If no language has been selected, TPIPE uses the Windows internal default mapping.

7 – Convert to Title Case

Converts all text to Title Case -- i.e., the first letter of every word is capitalized, and all other letters are forced to lower case. This routine calculates a table of upper and lower case letters on TPIPE startup, and this determination is based on the semantics of the language selected in Control Panel.

8 – Convert to Sentence Case

Converts all text to Sentence case ie the first word in every sentence is capitalized, all other letters are left as is. Sentences start after periods, exclamation marks, colons, question marks, quotes, parentheses and angle brackets (.!:?"<()).

9 – Convert to tOGGLE cASE

tOGGLES tHE cASE of all text -- i.e., all UPPERCASE characters are converted to lowercase and vice-versa.

10 – Remove blank lines

Removes blank lines. Note, lines with spaces or tabs are not removed. Use the Remove Blanks From Start Of Line filters first to rectify this.

11 – Remove blanks from End of Line

Removes spaces and tabs from the end of every line.

12 – Remove blanks from Start of Line

Removes spaces and tabs from the start of every line.

13 – Remove binary characters

Removes binary characters such as those higher than ASCII code 127, and those less than ASCII code 32 except for carriage returns (ASCII code 13) and line feeds (ASCII code 10).

This filter is very useful if you have a corrupted text file, or if you just want to see what text is inside a binary file. The binary information is removed, leaving you with just the text.

14 – Remove ANSI codes

ANSI (American National Standards Institute) codes are included in various streams of information, to provide a remote computer with control over cursor positioning, text attributes, etc. They are also used in connections between minicomputers and mainframe computers and the terminals connected to them.

The need to use an ANSI filter can be recognized when something like the following example shows up in a file viewed in a text editor:

```
<[0;1;4mas<[m - MC88000 assembler
```

In this example, the "as" near the beginning is displayed in a different color than the rest of the line when the ANSI codes are properly processed. The Escape (ASCII 27) codes above have been replaced by the < symbol to make this line printable.

The Remove ANSI Escape Sequences filter can be used to filter out these codes and "clean up" the text so that it can be used in standard fashions such as copying and pasting into a word processor. On Unix machines the man (manual) help utility will only allow page-by-page browsing through a file in a forward direction. By piping the man output to a text file, transferring it to a DOS machine, and running it through the Remove ANSI Escape Sequences filter (and the Convert EOL filter - Unix to DOS if desired), a standard DOS editor can be used for browsing through the file, quoting from it, etc.

15 – Convert IBM drawing characters

IBM drawing characters in the upper ASCII range (128-255) are commonly used to draw lines and boxes, single and double line borders, shaded characters etc.

Many devices (such as printers, non-IBM computers etc.) do not support the display of these characters.

This filter converts them to standard ASCII characters (+, - and |) that all computers can display.

16 – Remove HTML and SGML

Use this filter to convert HTML documents to a readable format. This filter removes HTML and XML markup tags i.e. everything including and between <> brackets.

17 – Remove backspaces

Remove backspaces, i.e. all ASCII code 8's.

18 – Resolve backspaces

Resolve backspaces -- i.e., remove both the backspaces and the characters prior to the backspaces that would have been deleted.

- 19 – Remove multiple whitespace
Removes sequences of multiple spaces or tabs and replaces them with a single space.
- 20 – UUEncode
Usually used for transmitting binary files inside an email. Files of this type are usually given an extension of **.uue**. Warning – UUencoded text may be corrupted when passing over a mainframe mail gateway. To avoid corruption, use Mime Base 64 or XXEncode.
- 21 – Hex Encode
A very simple encoding of a file. Usually used for small files, because it uses a large amount of space. The benefit is that the file is very easy to encode/decode, and the file cannot be corrupted passing through mail gateways.
- 22 – Hex Decode
Converts a file from its hex representation back to binary. The file to be decoded MUST NOT have any extra characters at the start or end if it is to be successfully processed.
- 23 – MIME Encode (Base 64)
Used for binary data. Files of this type are usually given an extension of **.b64**.
- 24 – MIME Decode (Base 64)
Used for binary data. Files of this type are usually given an extension of **.b64**. The file to be decoded MUST NOT have any extra characters at the start or end if it is to be successfully processed.
- 25 – MIME Encode (Quoted printable)
Quoted printable is used for text that is mainly readable, but may contain special characters with accents etc.
- 26 – MIME Decode (Quoted printable)
The inverse of the above encoding.
- 27 – UUDecode
Mail attachments can be uuencoded, use this filter to convert the file back to its correct form. Files of this type are usually given an extension of **.uue**.
- 28 – Extract email addresses
Extract email addresses. This filter searches for email addresses of the form `user@server.domain`, and writes them out one per line (using a DOS line feed, CR/LF). Usually this filter is followed by a filter to remove duplicate lines, and then by a Search and Replace filter, searching for \013\010 and replacing with a comma or semi-colon, depending on the email address separator used by your email software.
- 29 – Unscramble (ROT13)
This is a simple email encoding usually used to disguise text that some people may find offensive. The encoding is totally reversible (applying it twice removes the encoding). Only alpha characters are affected (A..Z and a..z).
- 30 – Hex dump
This changes the text to lines consisting of 16 bytes each. Each line has an 8 hex digit file index, 16 bytes (in hex) and the ASCII representation:
- ```
00000000 65 67 69 6E 0D 0A 20 20 20 20 20 61 64 64 72 egin.....addr
00000010 65 73 73 20 3A 3D 20 0D 0A 20 20 20 20 20 20 ess.:=.....
00000020 20 64 65 63 54 6F 48 65 78 53 74 72 28 20 28 66 .decToHexStr.(f
```
- This filter is very useful for identifying special characters to search and replace.
- 32 – XXEncode  
Essentially identical to UUEncode except that the character set used is different to allow it to pass through EBCDIC gateways without corruption. The XXencoding implemented by TPIPE uses the following characters:  
+-

0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

33 – XXDecode

Essentially identical to UUDecode except that the character set used is different to allow it to pass through EBCDIC gateways without corruption.

34 – Reverse line order

The order of the input lines is reversed i.e. the last line comes out first and the first line comes out last. A file is read entirely into RAM before being reversed, so be wary of reversing files that are larger than your machine's RAM size.

35 – Remove email headers

This filter removes the email headers that accompany emails exported to a text format. The email headers are the lines such as To:, From:, Subject: and various other message headers added by all the servers through which your email passes before it gets to its destination.

36 – Decimal dump

This changes the text to lines consisting of 10 bytes each. Each line has a 10 decimal digit file index, 10 bytes (in decimal) and the ASCII representation:

0000000000 080 108 101 097 115 101 032 102 101 101 Please fee  
0000000010 108 032 102 114 101 101 032 116 111 032 I free to  
0000000020 099 111 109 109 101 110 116 013 010 111 comment..o

This filter is very useful for identifying special characters to search and replace.

37 – HTTP Encode

This filter is used to encode text for use in an HTTP header – a (usually) small piece of text that accompanies a web page request to a web server. This filter is very useful for debugging CGI scripts because it can create HTTP requests in the correct form. HTTP encoded text usually looks like the following:

a+%28usually%29+small+piece+of+text+that+accompanies+a+web+page  
+request+to+a+web+server.+This+filter+is+very+

38 – HTTP Decode

This filter is used to decode text from an HTTP header – a (usually) small piece of text that accompanies a web page request to a web server.

39 – Randomize lines

This filter puts lines into random order. This is useful when a random sample of data is required for statistical purposes - just follow this filter with a head/tail of file filter (/head or /tail). The lines output will differ from one run to the next; the order is determined by a pseudo-random number generator.

40 – Create word list

This filter takes all the incoming words and outputs them one per line. This can be used to generate word lists for indexes, encryption programs etc. Hyphenated words are recognized as single words, provided that they aren't broken across lines. To get around this limitation, use a Search and Replace filter to replace hyphens followed by line feeds with just a hyphen. Normally you would follow this filter with a remove duplicates filter, or alternatively, a Count Duplicate Lines filter (with Include counts of 1).

catch22 – a word

24-7 – a word

twenty-four – a word

5th – a word

ice cream – two words

Commas or periods after words are treated as word separators.

41 – Reverse each line

Each line is output reversed from left to right. This can be useful to extract domain names from web site log files - use this filter to reverse each line, use an extract

matches filter of `[w\|d]+.\[w\|d]+` to extract each domain name, then reverse each line again. Note: This filter will NOT work on Unicode or UTF-8 data. It will only work on single-byte data such as ASCII or ANSI.

#### 42 – Convert to RanDOM case

This filter randomly changes the case of characters. This routine calculates a table of upper and lower case letters on TPIPE startup, and this determination is based on the semantics of the language selected in the Windows Control Panel. Running this filter again will generate different results; for example:

1. ranDoMize cASe
2. RanDOmIZE case
3. randomIZE casE

#### 43 – Extract URLs

Extract URLs. This filter lists `http://`, `https://`, `ftp://` and `gopher://` URLs one per line.

#### 44 – ANSI to Unicode

Converts single byte ANSI characters to double byte Unicode characters. This filter can be useful if you want to send a text file to someone using a language other than your own. This filter is often followed by an Add Header filter, to add a Unicode byte order mark (BOM), `\xFF\xFE`.

#### 45 – Unicode to ANSI

Converts double byte Unicode characters to single byte ANSI characters. This filter can be useful if you want to send a text file to someone using a language other than your own. This filter is often followed by a Remove start or end of file filter, to either remove the first two bytes of Unicode (before the conversion) or the first byte of ANSI (after the conversion), to remove the leading Unicode byte order mark (BOM).

#### 46 – Display debug window

A debug filter is very handy for debugging filters. When text is passed through this filter, it places the output into a window so that you can see what the text looks like at that stage of the filtering process.

#### 47 – Word concordance

This filter generates a word concordance. A word concordance shows the context or surrounding words for a given set of words in a dictionary.

#### 48 – Remove all

This filter removes all text. Unlike a pattern match filter that matches everything and then throws it away, this filter is far more efficient, especially for large files, as it signals completion back to the input filter so only the first chunk of a multi-gigabyte file will ever get processed.

It is useful in two main situations

1. Inside a subfilter, it prevents any of the subfiltered text from re-entering the text stream. So you could restrict to lines matching a pattern, output the matching lines to a new file, and then remove them.
2. To remove all of the text of a file, then use an Add Header filter with the `@fullInputFilename` macro to obtain the name of the file.

Note: An Add Left Margin or Add Right Margin filter will not work after a Remove All filter, as they require an actual line to trigger them. Instead, use an Add Header or Add Footer filter.

#### 49 – Restrict to each line in turn

This filter restricts sub filters to operate on each line in turn. This filter is used for its side effect of limiting the matched text to a single line at most.

#### 50 – Convert CSV to Tab-delimited

Converts CSV data (quoted or unquoted) to tab-delimited form. It's preferable to use a file with column headers, because then TPIPE can easily determine if the

fields have embedded CR/LFs in them. If the data is properly quoted then TPIPE will determine this automatically.

51 – Convert CSV to XML

Converts CSV data (quoted or unquoted) to XML form. It's preferable to use a file with column headers, because then TPIPE can easily determine if the fields have embedded CR/LFs in them. If the data is properly quoted then TPIPE will determine this automatically. TPIPE correctly escapes < > " ' and & in the data to the corresponding XML entity. If your data contains invalid XML characters such as ASCII 26 (End-of-file, hex \x1A), follow this filter with a search/replace filter to remove \x1A and replace with nothing.

52 – Convert Tab-delimited to CSV

Converts Tab-delimited data to CSV data. It's preferable to use a file with column headers, because then TPIPE can easily determine if the fields have embedded CR/LFs in them. TPIPE cannot determine this without column headers.

53 – Convert Tab-delimited to XML

Converts Tab-delimited data to XML data. It's preferable to use a file with column headers (/simple=55), because then TPIPE can easily determine if the fields have embedded CR/LFs in them. TPIPE cannot determine this without column headers. TPIPE correctly escapes < > " ' and & in the data to the corresponding XML entity. If your data contains invalid XML characters such as ASCII 26 (End-of-file, hex \x1A), follow this filter with a search/replace filter to remove \x1A and replace with nothing.

54 – Convert CSV (with column headers) to XML

See description for 51 – Convert CSV to XML.

55 – Convert Tab-delimited (with column headers) to XML

See description for 53 – Convert Tab-delimited to XML.

56 – Convert CSV (with column headers) to Tab-delimited

See description for 50 – Convert CSV to Tab-delimited.

57 – Convert Tab-delimited (with column headers) to CSV

See description for 52 – Convert Tab-delimited to CSV.

58 – Restrict to file name

This filter applies its subfilters only to files with filenames (ie drive + path + filename) matching or not matching a pattern or list of patterns. This is very handy for only applying a Convert Word Documents to Text filter only to files matching the pattern

\.DOC\$

With the appropriate pattern, this filter can also be used to control subfilters based on filename, folder and drive. Note that this filter uses case-insensitive Perl regular expressions, not Windows wildcards.

59 – Convert Word documents to text

This filter takes ALL incoming documents, opens them with Microsoft Word, and outputs them as text files. This can be used to process a set of Word Documents to text file format. After this filter you can add search and replace filters or any other filters you choose.

This filter requires Microsoft Word 98 or higher to be installed. If you wish to convert documents other than the default .DOC files, you may also need to install Word's conversion filters. If Word cannot be started automatically TPIPE will prompt you to start it manually before continuing.

Unless you know that all documents being processed are Word documents (e.g. by using a wildcard of \*.doc in the Files to Process tab), you should restrict this filter to only files matching the pattern:

\.DOC\$

60 – Swap UTF-16 word order

This filter swaps pairs of bytes

e.g.

| Byte number | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  |
|-------------|----|----|----|----|----|----|----|----|
| Input File  | FF | FE | 00 | 20 | 00 | 31 | 00 | 32 |
| Output File | FE | FF | 20 | 00 | 31 | 00 | 32 | 00 |

This is commonly used to transform big-endian or little-endian Unicode files so that other programs can use them.

#### 61 – Swap UTF-32 word order

This filter swaps groups of 2-byte words.

e.g.

| Byte number | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  |
|-------------|----|----|----|----|----|----|----|----|
| Input File  | FF | FE | 00 | 00 | 00 | 31 | 00 | 00 |
| Output File | 00 | 00 | FE | FF | 00 | 00 | 31 | 00 |

This is commonly used to transform big-endian or little-endian Unicode files so that other programs can use them.

#### 62 – Remove BOM (Byte Order Mark)

This filter removes the Unicode Byte Order Mark from the start of Unicode files, if it is present.

| Bytes removed | Description           |
|---------------|-----------------------|
| 00 00 FE FF   | UTF-32, big-endian    |
| FF FE 00 00   | UTF-32, little-endian |
| FE FF         | UTF-16, big-endian    |
| FF FE         | UTF-16, little-endian |
| EF BB BF      | UTF-8                 |

#### 63 – Make Big Endian

Converts a Little Endian Unicode file into a Big Endian Unicode file

e.g.

| Input file              | Output file             |
|-------------------------|-------------------------|
| 00 00 FE FF 00 00 00 4D | Unchanged               |
| FE FF 4E 8C             | Unchanged               |
| FF FE 00 00 4D 00 00 00 | 00 00 FE FF 00 00 00 4D |
| FF FE 8C 4E             | FE FF 4E 8C             |

Note - the file MUST start with a Byte Order Mark (BOM) for it to be correctly identified.

#### 64 – Make Little Endian

Converts a Big Endian Unicode file into a Little Endian Unicode file

e.g.

| Input file              | Output file             |
|-------------------------|-------------------------|
| 00 00 FE FF 00 00 00 4D | FF FF 00 00 4D 00 00 00 |
| FE FF 4E 8C             | FF FE 8C 4E             |
| FF FE 00 00 4D 00 00 00 | Unchanged               |
| FF FE 8C 4E             | Unchanged               |

Note - the file MUST start with a Byte Order Mark (BOM) for it to be correctly identified.

#### 65 – Compress to Packed Decimal

This filter compresses EBCDIC numeric data (optional leading sign, numbers and periods) to an EBCDIC packed decimal field (also known as Comp-3).

There are several notes to keep in mind when using this filter:

1. You MUST use this filter inside a Restrict to Byte Range filter. The field WIDTH is then set by the containing filter.
2. Compressing a field will decrease your output record length, so ensure you allow for this. A good strategy to avoid problems is to first compress the rightmost field, then work your work back to the leftmost field. This prevents the field column positions from changing and makes the file easier to work with.

This filter will add hex 'B' to negative fields, hex 'C' to positive fields and hex 'F' to unsigned fields. If these codes don't match what your target needs, use a column or CSV restriction to apply a search/replace.

#### 66 – Compress to Zoned Decimal

This filter expands an EBCDIC zoned decimal field to a raw EBCDIC number with a sign. Typically this filter is then followed by a Convert EBCDIC to ASCII filter - after all other fields have been expanded as well.

There are several notes to keep in mind when using this filter:

1. You MUST use this filter inside a Restrict to Byte Range filter. The field WIDTH is then set by the containing filter.
2. Expanding a field will increase your output record length, so ensure you allow for this. A good strategy to avoid problems is to first expand the rightmost field, then work your work back to the leftmost field. This prevents the field column positions from changing and makes the file easier to work with.

#### 67 – Expand Binary Number to EBCDIC

This filter expands a series of digits stored in binary (BIG ENDIAN) form. The maximum width is 8 bytes.

There are several notes to keep in mind when using this filter:

1. You MUST use this filter inside a Restrict to Byte Range filter. The field WIDTH is then set by the containing filter.
2. Expanding a field will increase your output record length, so ensure you allow for this. A good strategy to avoid problems is to first expand the rightmost field, then work your work back to the leftmost field. This prevents the field column positions from changing and makes the file easier to work with.
3. If the data is stored in LITTLE ENDIAN order, use a Reverse filter inside the Restriction prior to the Expand Binary Numbers filter.

#### 68 – Expand Binary Number to ASCII

This filter expands a series of digits stored in binary (BIG ENDIAN) form. The maximum width is 8 bytes.

There are several notes to keep in mind when using this filter:

1. You MUST use this filter inside a Restrict to Byte Range filter. The field WIDTH is then set by the containing filter.
2. Expanding a field will increase your output record length, so ensure you allow for this. A good strategy to avoid problems is to first expand the rightmost field, then work your work back to the leftmost field. This prevents the field column positions from changing and makes the file easier to work with.
3. If the data is stored in LITTLE ENDIAN order, use a Reverse filter inside the Restriction prior to the Expand Binary Numbers filter.

#### 69 – NFC - Canonical Decomposition, followed by Canonical Composition

Applies a Unicode NFC - Canonical Decomposition, followed by Canonical Composition transformation to incoming Unicode text (UTF16-LE). Output is also Unicode UTF16-LE.

**70 – NFD - Canonical Decomposition**

Applies a Unicode NFD - Canonical Decomposition transformation to incoming Unicode text (UTF16-LE). Output is also Unicode UTF16-LE.

**71 – NFKD - Compatibility Decomposition**

Applies a Unicode NFKD - Compatibility Decomposition transformation to incoming Unicode text (UTF16-LE). Output is also Unicode UTF16-LE.

**72 – NFKC - Compatibility Decomposition, followed by Canonical Composition**

Applies a Unicode NFKC - Compatibility Decomposition, followed by Canonical Composition transformation to incoming Unicode text (UTF16-LE). Output is also Unicode UTF16-LE.

**73 – Decompose****74 – Compose**

Applies a Unicode Compose transformation to incoming Unicode text (UTF16-LE). The output is also Unicode UTF16-LE.

**75 – Convert numeric HTML Entities to text**

This filter converts decimal/hex numeric HTML/XML entities to plain text. For example:

&#174; → ®

&#xAE; → ®

Typically, the input file is ANSI (single byte) format. This filter will output UTF-8 characters for high-value entities e.g. &#6144;. The best approach is to first convert the file from ANSI to UTF-8 (/unicode), then apply this filter.

**76 – Convert PDF documents to text**

This filter takes ALL incoming documents and converts them from PDF to text.

Most of the formatting will be lost.

**77 – Restrict to ANSI files****78 – Restrict to Unicode UTF16 files****79 – Restrict to Unicode UTF32 files****80 – Convert Excel spreadsheets to text**

This filter takes ALL incoming documents, opens them with Microsoft Excel, and outputs them as CSV (comma-delimited) files. After running this filter, you can add search and replace filters or any other filters you choose, such as convert the data to Tab-delimited or XML.

This filter requires Microsoft Excel 98 or higher to be installed. If you wish to convert documents other than the default .XLS files, you may also need to install Excel's conversion filters.

Unless you know that all documents being processed are Excel documents (e.g. by using a wildcard of \*.xls in the Files to Process tab), you should restrict (/simple=58) this filter to only files matching the pattern

.XLS\$

**81 - Shred file****82 - Unicode to escaped ASCII****83 - Restrict to Unicode files****84 - T-filter**

The T-Filter allows you to process the same output in multiple ways. You can create a subfilter, and add filters to create the desired output. When this side of the T has finished processing, the data is discarded and the original text continues processing as though the T-filter did not exist.

**85 - Convert decimal/hex numeric HTML/XML entities and entity names to text (i.e., &#174; -> ®, or &reg; -> ®).** This filter outputs UTF-8 characters for high-value entities.

**/SORT=Type,Reverse,RemoveDuplicates,StartColumn,Length**

Sort text files. Note that /SORT is slow on large files; it's intended for simple sorts of relatively small files. For big files, you should use a dedicated sorting application.

**Type** - the sort type

- 0 - ANSI sort
- 1 - ANSI sort (case sensitive)
- 2 - ASCII sort
- 3 - ASCII sort (case sensitive)
- 4 - Numeric sort
- 5 - Sort by length of line

**Reverse** - If 1, sort in descending order; if 0, sort in ascending order

**RemoveDuplicates** - If 1, remove duplicate lines; if 0 keep duplicate lines

**StartColumn** - The column in the line to begin the comparisons

**Length** - The length of the comparison

**/split=type,SplitSize,SplitChar,SplitCharPos,SplitCharCount,SplitLines,SplitFilename**

Adds a split type filter. The arguments are:

**type:**

- 0 Split at a given size
- 1 Split at a given character
- 2 Split at a given number of lines

**splitSize** - the size file to split at

**splitChar** - the character to split at

**splitCharPos**

- 0 Split before the character (it goes into the next file)
- 1 Split after the character (it remains in the first file)
- 2 Split on top of the character (remove it)

**SplitCharCount** - the number of times to see SplitChar before splitting

**SplitLines** - (optional) split after a given number of lines, default 60

**SplitFilename** - (optional) the name to give to each output split file. /split will append a "%3.3d" format specifier to the name; i.e. SplitFilename of "foo.txt" will generate output files named "foo.txt.000", "foo.txt.001", etc. If you don't specify a SplitFilename, /split will use the input filename as the base.

**/string=type,MatchCase,string**

Add a string-type filter. The arguments are:

**type:**

- 0 - Add left margin
- 1 - Add header
- 2 - Add footer
- 3 - Add right margin
- 4 - Remove lines that match exactly
- 5 - Retain lines that match exactly
- 6 - Remove lines matching the Perl pattern
- 7 - Retain lines matching the Perl pattern
- 8 - Add text side by side
- 9 - Add repeating text side by side
- 10 - Not Used
- 11 - Not Used
- 12 - XSLT transform
- 13 - Restrict to lines from list
- 14 - Restrict to lines NOT in list
- 15 - Restrict to lines matching the Perl pattern
- 16 - Restrict to lines NOT matching the Perl pattern
- 17 - Restrict to filenames patching the Perl pattern
- 18 - Restrict to filenames NOT matching the Perl pattern

**matchCase** - case sensitive or not (where appropriate)

**string** - the string to use

#### **/tail=Exclude,LinesOrBytes,Count**

Add a tail type filter (includes or excludes text at the end of the file). The arguments are:

##### **Exclude:**

- 0 - Include the text
- 1 - Exclude the text

##### **LinesOrBytes:**

- 0 - Measure in lines
- 1 - Measure in bytes

**Count** - the number of lines or bytes to include or exclude

#### **/unicode=input,output**

Convert the file to or from Unicode. **input** is the encoding for the input file; **output** is the encoding for the output file. The possible values are:

UTF-16LE  
UTF-16BE  
UTF-32LE  
UTF-32BE  
UTF-8  
ANSI  
ASCII

CP $nnn$ , where  $nnn$  is a Windows code page (for example, CP437 or CP1251).

TPIPE handles files internally as UTF-8, so if you want to process a Windows UTF-16LE file,

you'll need to convert it to UTF-8 first, then apply the desired filters, and convert it back to UTF-16LE. For example, to wrap a Unicode file at column 80:

```
tpipe /input=inputname /output=outputname /unicode=UTF-16LE,UTF-8 /number=2,80 /
unicode=UTF-8,UTF-16LE
```

**/xml=Type,IncludeText,IncludeQuotes,MatchCase,BufferSize,Tag,Attribute,EndTag**

Adds an HTML / XML filter. The arguments are:

**Type** - the operation to perform:

- 0 restrict to an element
- 1 restrict to an attribute
- 2 restrict to between tags

**IncludeText** - whether to include the find string in the restriction result (default false)

**IncludeQuotes** - whether to include surrounding quotes in the attribute result or not (default false)

**MatchCase** - match case exactly or not (default false)

**BufferSize** - the maximum expected size of the match (default 32768)

**Tag** - the element or start tag to find

**Attribute** - the attribute to find

**EndTag** - the endTag to find

#### 4.3.159 TRANSIENT

**Purpose:** Toggle the shell's transient mode

**Format:** TRANSIENT [on | off]

**Usage:**

TRANSIENT allows you to change the shell's transient mode (i.e., whether it was started with a /C), so that you can make a transient session permanent (or vice versa).

#### 4.3.160 TREE

**Purpose:** Display a graphical directory tree

**Format:** TREE [[/A:[[-+]]rhsadecijopt /A /B /D /F /H /Nj /O:[-]adegnrstu /P /S[n] /T[:a|c|w] /Z ] dir...]

**dir** The directory to use as the start of the tree. If one or more directories are specified, TREE will display a tree for each specified directory. If none are specified, the tree for the current working directory is displayed.

[/A: \(Attribute select\)](#)  
[/A\(SCII\)](#)

[/O\(rder\)](#)  
[/P\(ause\)](#)

/B(are)  
/D(escriptions)  
/F(iles)  
/HIDDEN directories  
/N (disable option)

/S (file size)  
/Sn (subcategory depth)  
/T(ime and date)  
/Z (file size)

#### **File Selection:**

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#) (with [/F](#)), and [multiple file names](#).

#### **Usage:**

The TREE command displays a graphical representation of the directory tree using standard or extended ASCII characters. For example, to display the directory structure on drive C:

```
[c:\] tree c:\
```

TREE uses the standard line drawing characters in the U.S. English extended ASCII character set. If your system is configured for a different country or language, or if you use a font which does not include these line drawing characters, the connecting lines in the tree display may not appear correctly (or not appear at all) on your screen. To correct the problem, use [/A](#), or configure the **TCC** to use a font which can display standard extended ASCII characters.

You can print the display, save it in a file, or view it with [LIST](#) by using standard [redirection](#) symbols. Be sure to review the [/A](#) option before attempting to print the TREE output. The options discussed below specify the amount of information included in the display.

#### **Options:**

- /=** Display the TREE command dialog to help you set the command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)
- /A** Display the tree using standard ASCII characters. You can use this option if you want to save the directory tree in a file for further processing or print the tree on a printer which does not support the graphical symbols that TREE normally uses.
- /A:[..]** Select only those files that match the specified attribute(s). See [Attribute Switches](#) for details.  
  
You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /B** Display the full pathname of each directory, without any of the line-drawing characters.
- /D** Display file and directory descriptions.
- /F** Display files as well as directories. If you use this option, the name of each file is displayed beneath the name of the directory in which it resides.
- /H** Display hidden as well as normal directories. If you combine **/H** and **/F**, hidden files are also displayed.

- /N** Disables the specified options:
- j** Skip junctions
- /O:....** Sort the files before processing. You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:
- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
  - Reverse the sort order for the next sort key
  - a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
  - d** Sort by date and time (oldest first); also see **/T:acw**
  - e** Sort by extension
  - g** Group subdirectories first, then files
  - r** Reverse the sort order for all options
  - s** Sort by size
  - t** Same as **d**
  - u** Unsorted
- /P** Wait for a key to be pressed after each screen page before continuing the display. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /S** If you specify a number after the **/S**, TREE will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\e", **/S2** will only affect the "a", "b", and "c" directories. If you do not specify a number, **/S** shows the file sizes (see [/Z](#)).
- /T** Display the time and date for each directory. If you combine **/T** and [/F](#), the time and date for each file will also be displayed.
- By default, the time and date shown will be of the last modification. You can select a specific time and date stamp by using the following variations of **/T**:
- /T:a** Last access date and time (access time is not displayed on VFAT and FAT32 volumes).
  - /T:c** Creation date and time.
  - /T:w** Last modification ("write") date and time (default).
- /Z** Display the size of each file. **/Z** without a **/F** will display the subdirectory tree sizes (the size of the current directory and all of its subdirectories).

#### 4.3.161 TRUENAME

**Purpose:** Find the full, true path and file name for a file

**Format:** TRUENAME file

See also: The [@TRUENAME](#) variable function.

**Usage:**

Network reassignments, junctions, symbolic links, and the SUBST command can obscure the true name

of a file. TRUENAME "sees through" these obstacles and reports the fully qualified name of a file.

The following example uses TRUENAME to get the true pathname for a file:

```
[c:\] subst d: c:\util\test
[c:\] truename d:\test.exe
c:\util\test\test.exe
```

### 4.3.162 TYPE

**Purpose:** Display the contents of the specified file(s)

**Format:** TYPE [/A:[[-][+]rhsadecijopt] /B /"text" /L /O:[-]adegnrstu /P /X /XS] [[@file](#)] *file...*

**file** The file or list of files that you want to display.

**@file** A text file containing the names of the files to display, one per line (see [@file lists](#) for details).

[/A: \(Attribute select\)](#)

[/B\(ell\)](#)

[/"/text" \(match description\)](#)

[/L\(ine numbers\)](#)

[/P\(ause\)](#)

[/O\(rder\)](#)

[/X\(hex\)](#)

[/XS \(hex w/spaces\)](#)

See also: [HEAD](#), [TAIL](#), [LIST](#).

#### File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

**Internet:** Can be used with [FTP and HTTP servers](#), e.g.

```
type "http://jpsoft.com/notfound.htm"
```

#### Usage:

The TYPE command displays a file. It is normally only useful for displaying text files (i.e. alphanumeric characters arranged in lines separated by CR/LF). Most text files use either ASCII or Unicode.

Executable files (.EXE) and many data files may be unreadable when displayed with TYPE because they include non-alphanumeric characters or unusual line separators.

To display the files *MEMO1* and *MEMO2*:

```
type /p memo1 memo2
```

You can press **Ctrl-S** to pause TYPE's display and then any key to continue.

To display text from the clipboard use **CLIP:** as the file name. CLIP: will not return any data if the clipboard does not contain text. See [Redirection](#) for more information on CLIP:.

You will probably find LIST to be more useful for displaying files on the screen. The TYPE /L command used with [redirection](#) is useful if you want to add line numbers to a file, for example:

```
type /l myfile > myfile.num
```

TYPE sets two internal variables:

|               |                             |
|---------------|-----------------------------|
| %_type_files  | The number of files renamed |
| %_type_errors | The number of errors        |

- **NTFS File Streams**

TYPE supports file streams on NTFS drives. You can type an individual stream by specifying the stream name, for example:

```
type streamfile:s1
```

See [NTFS File Streams](#) for additional details.

**Options:**

**/=** Display the TYPE command dialog to help you set the command line options. You cannot specify any other arguments on the command line. (Not available in **TCC/LE**.)

**/A:** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow **/A:**. Do not use **/A:** with @file lists. See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

**/B** Ignore bell (ASCII 7) characters.

**/I"text"** Select files by matching text in their descriptions. The text can include [wildcards](#) and extended wildcards. The search text must be enclosed in double quotes, and must follow the **/I** immediately, with no intervening spaces. You can select all filenames that have a description with **/I"[?]\*"**, or all filenames that do not have a description with **/I"[]"**. Do not use **/I** with @file lists. See [@file lists](#) for details.

**/L** Display a line number preceding each line of text.

**/O:...** Sort the files before processing. (Not available in **TCC/LE**.)

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size

**t** Same as **d**  
**u** Unsorted

- /P** Prompt after displaying each page. Your options at the prompt are explained in detail under [Page and File Prompts](#).
- /X** Display the file in hex. (Not available in **TCC/LE**.)
- /XS** Display the file in hex, using spaces instead of periods for non-printable characters. (Not available in **TCC/LE**.)

### 4.3.163 UNALIAS

**Purpose:** Remove aliases from the alias list

**Format:** UNALIAS [/Q /R *file...* (*alias ...*)] *alias...*  
or  
UNALIAS \*

**alias** One or more aliases to remove from memory.  
**file** One or more files from which to read the aliases to be undefined.

[/Q\(uiet\)](#)                            [/R\(ead file\)](#)

See also: [ALIAS](#) and [ESET](#).

**Usage:**

**TCC** maintains a list of the aliases that you have defined. The UNALIAS command will remove aliases from that list. UNALIAS supports wildcards in the alias name.

For example, to remove the alias DDIR:

```
unalias ddir
```

To remove all the aliases:

```
unalias *
```

To remove all the aliases that begin with "DD":

```
unalias dd*
```

You can delete all matching aliases except for those specified by enclosing the exceptions in parentheses. For example, to remove all aliases beginning with "a" except for *alias1* and *alias2*:

```
unalias (alias1 alias2) a*
```

If you keep aliases in a file that can be loaded with the [ALIAS /R](#) command, you can remove the aliases by using the UNALIAS /R command with the same file name:

```
unalias /r alias.lst
```

This is much faster than removing each alias individually in a batch file, and can be more selective than using UNALIAS \*. UNALIAS /R accepts all of the alias definition formats you can use in a file for ALIAS /R.

### *Options:*

- /Q** Prevents UNALIAS from displaying an error message if one or more of the aliases does not exist. This option is most useful in batch files, for removing a group of aliases when some of the aliases may not have been defined.
  - /R** Read the list of aliases to remove from a file. The file format should be the same format as that used by the [ALIAS /R](#) command. You can use multiple files with one UNALIAS /R command by placing the names on the command line, separated by spaces:

```
unalias /r alias1.lst alias2.lst
```

UNALIAS /R will read from stdin if no filename is present and input is redirected.

### 4.3.164 UNFUNCTION

**Purpose:** Remove user-defined functions from the function list

**Format:** UNFUNCTION [/G /L /Q /R file... (function ...)] function...  
or  
UNFUNCTION \*

|                 |                                                                 |
|-----------------|-----------------------------------------------------------------|
| <b>function</b> | One or more functions to remove from memory.                    |
| <b>file</b>     | One or more files from which to read functions to be undefined. |

**/Q(quiet)**                                   **/R(read file)**

See also: [FUNCTION](#) and [ESET](#).

### ***Usage:***

**TCC** maintains a list of the functions that you have defined. The UNFUNCTION command will remove functions from that list. UNFUNCTION supports wildcards in the function name.

To remove the function DDIR:

```
unfunction ddir
```

To remove all the functions:

```
unfunction *
```

To remove all the functions that begin with "DD":

unfunction dd\*

You can delete all matching functions except for those specified by enclosing the exceptions in parentheses. For example, to remove all functions beginning with "f" except for *func1* and *func2*:

```
unfunction (func1 func2) #*
```

If you keep functions in a file that can be loaded with the [FUNCTION /R](#) command, you can remove the functions by using the UNFUNCTION /R command with the same file name:

```
unfunction /r function.lst
```

This is much faster than removing each function individually in a batch file, and can be more selective than using UNFUNCTION \*.

#### **Options:**

- /G** Remove the function(s) from the global list.
- /L** Remove the function(s) from the local list.
- /Q** Prevents UNFUNCTION from displaying an error message if one or more of the functions does not exist. This option is most useful in batch files, for removing a group of functions when some of the functions may not have been defined.
- /R** Read the list of functions to remove from a file. The file format should be the same format as that used by the [FUNCTION /R](#) command. You can use multiple files with one UNFUNCTION /R command by placing the names on the command line, separated by spaces:

```
unfunction /r function1.lst function2.lst
```

UNFUNCTION /R will read from stdin if no filename is present and input is redirected.

## **4.3.165 UNGZIP**

**Purpose:** Add, update, or delete files in a .gz (GZIP) archive

**Format:** GZIP [/A:[[-][+]rhsdaecjot] /E /O /Q /V] [gziparchive] path

**gziparchive** The gzip file to work with  
**path** The path where files will be extracted

|                                           |                          |
|-------------------------------------------|--------------------------|
| <a href="#">/A:... (attribute switch)</a> | <a href="#">/Q(uiet)</a> |
| <a href="#">/E(xtract)</a>                | <a href="#">/V(iew)</a>  |
| <a href="#">/O(verwrite)</a>              |                          |

See also [GZIP](#).

#### **File Selection**

Supports [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

#### **Usage:**

UNGZIP is compatible with the archives created by the Linux / UNIX gunzip utility, and supports RFC 1952.

You can specify a pathname for **gziparchive**. If you don't provide an extension, and the filename as

entered doesn't exist, GZIP adds ".gz". If you don't specify an operation, UNGZIP will default to Extract.

**Option:**

**/A:...** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with **@file** lists. See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)

**/E** Extract (default).

**/M** Delete the files from the disk after adding them to the gzip file.

**/O** Overwrite existing files.

**/Q** Don't display the filenames as they are extracted from the archive.

**/V** View the list of files in the zip file (date, time, and filename). Due to the limitations of the GZIP format, this can only display the first file in the archive. If the file was compressed with lzw, it will not have a header, so it cannot be viewed.

### 4.3.166 UNSET

**Purpose:** Remove variables from the environment or the registry

**Format:** UNSET [/D /E /Q /S /U /V /R file... (name ...)] name [name...]]]

or

UNSET \*

**name** One or more variables to remove (wildcards accepted except for registry variables).

**file** One or more files from which to read variables to be removed.

|                              |                           |
|------------------------------|---------------------------|
| <a href="#">/Default</a>     | <a href="#">/System</a>   |
| <a href="#">/Environment</a> | <a href="#">/User</a>     |
| <a href="#">/Quiet</a>       | <a href="#">/Volatile</a> |
| <a href="#">/Read</a>        |                           |

See also: [ESET](#) and [SET](#).

**Usage:**

UNSET removes one or more variables from the environment or from the Windows Registry.

For example, to remove the environment variable **CMDLINE**:

```
unset cmdline
```

If you use the command **UNSET \***, all of the environment variables will be deleted:

```
unset *
```

You can delete all matching variables except for those specified by enclosing the exceptions in parentheses. For example, to remove all variables beginning with "v" except for *var1* and *var2*:

```
unset (var1 var2) v*
```

UNSET can be used in a batch file, in conjunction with the [SETLOCAL](#) and [ENDLOCAL](#) commands, to clear the environment of variables that may cause problems for applications run from that batch file.

For more information on environment variables, see the [SET](#) command and the general discussion of the [environment](#).

**Note:** You cannot use UNSET with [GOSUB variables](#).

Use caution when removing environment variables, and especially when using **UNSET** \*. Many programs will not work properly without certain environment variables; for example, **TCC** depends on PATH.

**Registry Variables:** Default, System, User, and Volatile registry variables can be manipulated with the UNSET command's [/D](#), [/S](#), [/U](#) and [/V](#) switches, respectively. To remove the variable from both the registry and from the local environment, use both the [/E](#) switch and the registry variable selection switch together. (You cannot use wildcards for the variable name.) For example, to remove the volatile variable *myvar* from both the registry and the local environment, use:

```
unset /v /e myvar
```

Use caution when directly removing registry variables as they may be essential to various Windows processes and applications.

#### **Options:**

- /D** Delete a default variable from the registry (HKCU\DEFAULT\Environment).
- /E** When used together with one of [/D](#), [/S](#), [/U](#), or [/V](#), unsets both the registry variable and the local environment variable.
- /Q** Prevents UNSET from displaying any error messages.
- /R** Read environment variables to be UNSET from a file. This is much faster than using multiple UNSET commands in a batch file, and can be more selective than **UNSET** \*. The file format may be the same as that used by the [SET](#) /R command (see [SET](#) for more details), or it could just be one variable per line, wildcards not processed.

**UNSET /R** will read from STDIN if no filename is present and input is redirected.

- /S** Delete a **system** variable from the registry (HKLM\System\CurrentControlSet\Control\Session Manager\Environment).
- /U** Delete a **user** variable from the registry (HKCU\Environment).
- /V** Delete a **volatile** variable from the registry (HKCU\Volatile Environment)

### 4.3.167 UNSETARRAY

**Purpose:** Remove array variables

**Format:** UNSETARRAY [/Q] name [name...]  
or  
UNSETARRAY \*

**name** One or more array variables to remove (wildcards accepted).

[/Q\(quiet\)](#)

See also: [SETARRAY](#).

**Usage:**

UNSETARRAY removes one or more array variables.

For example, to remove the array variable **ARRAY1**:

```
unsetarray array1
```

If you use the command **UNSETARRAY \***, all of the array variables will be deleted:

```
unsetarray *
```

You can delete all matching array variables except for those specified by enclosing the exceptions in parentheses. For example, to remove all array variables beginning with "v" except for *var1* and *var2*:

```
unsetarray (var1 var2) v*
```

For more information on array variables, see the [SETARRAY](#) command.

**Options:**

**/Q** Prevents UNSETARRAY from displaying an error message if one or more of the array variables do not exist. This option is most useful in batch files, for removing a group of arrays when some of the arrays may not have been defined.

### 4.3.168 UNTAR

**Purpose:** Extract files from .TAR archives

**Format:** UNTAR [/A:[[-][+]rhsdaecjot] /C /D /E /F /G /Net /O /P /Q /TEST /U /V] tararchive path file ...

**ziparchive** The .tar file to work with  
**path** The path where files will be extracted  
**file** The file(s) to extract

|                                           |                               |
|-------------------------------------------|-------------------------------|
| <a href="#">/A:... (attribute switch)</a> | <a href="#">/O(overwrite)</a> |
| <a href="#">/C(contents)</a>              | <a href="#">/P(Percent)</a>   |
| <a href="#">/D(rectory)</a>               | <a href="#">/Q(quiet)</a>     |
| <a href="#">/E(xtract)</a>                | <a href="#">/TEST</a>         |

[/F\(reshen\)](#)  
[/G\(zip\)](#)  
[/N \(defaults\)](#)

[/U\(pdate\)](#)  
[/V\(iew\)](#)

See also [TAR](#).

### File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

### Usage:

UNTAR is compatible with tar archives created in Linux / UNIX. Unlike .zip archives, .tar archives are not compressed unless you use the gzip option. If you don't specify any arguments, UNTAR will display its command dialog.

You can specify a pathname for *tararchive*. If you don't provide an extension, and the filename as entered doesn't exist, UNTAR adds ".tar". If you don't specify an operation, UNTAR will default to Extract.

UNTAR supports wildcards for the tar archive name and for the filenames to extract.

*path* specifies the path where files will be extracted. If *path* is not specified, files are extracted to the current directory.

UNTAR supports [gzip](#) decompression, and can be used to extract .tar.gz archives.

UNTAR sets two internal variables:

%\_untar\_files      The number of files extracted  
%\_untar\_errors      The number of errors

### Options:

- /=      Display the UNTAR command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.
- /A:...      Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with [@file](#) lists. See [@file lists](#) for details.  
  
You can specify /A:= to display a dialog to help you set individual attributes. (Not available in **TCC/LE**.)
- /C      Display (on standard output) the contents of a file in the tar archive.
- /D      Recreate the directory structure saved in the tar file.
- /E      Extract the specified file(s). (This is the default.)
- /F      Extract only those files that currently exist in the target folder, and which are older than the file in the zip archive.

- /G      Use Gzip decompression.
- /N      Disable one or more default behaviors:
  - e      Don't display errors.
  - t      Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*).
- /O      Overwrite existing files. UNTAR normally prompts before overwriting an existing file; /O will suppress the prompt.
- /P      Display the progress (0 - 100%) for each file as it is extracted.
- /Q      Don't display filenames as they are extracted.
- /TEST    Test the integrity of the TAR file (header and contents). Any errors will be displayed on STDERR.
- /U      Extract files which either don't exist in the target folder, or which are older than the file in the zip archive.
- /V      View the list of files in the archive (date, time, size, and filename).

### 4.3.169 UNZIP

**Purpose:** Extract files from .ZIP archives

**Format:** UNZIP [/A:[[-][+]rhsdaecjot] /C /CRC /D /E /F /I /Nt /P /O /Q /S"password" /TEST /U /V] *ziparchive path file ...*

|                   |                                        |
|-------------------|----------------------------------------|
| <b>ziparchive</b> | The Zip file to work with              |
| <b>path</b>       | The path where files will be extracted |
| <b>file</b>       | The file(s) to extract                 |

|                                            |                               |
|--------------------------------------------|-------------------------------|
| <a href="#">/A:.... (attribute switch)</a> | <a href="#">/O(overwrite)</a> |
| <a href="#">/C(ontents)</a>                | <a href="#">/P(ercent)</a>    |
| <a href="#">/CRC</a>                       | <a href="#">/Q(uiet)</a>      |
| <a href="#">/D(irectory)</a>               | <a href="#">/S"password"</a>  |
| <a href="#">/E(xtract)</a>                 | <a href="#">/TEST</a>         |
| <a href="#">/F(reshen)</a>                 | <a href="#">/U(pdate)</a>     |
| <a href="#">/I (descriptions)</a>          | <a href="#">/V(iew)</a>       |
| <a href="#">/N</a>                         |                               |

#### File Selection

Supports [command dialog](#), [attribute switches](#), [extended wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

#### Usage:

You can specify a pathname for *ziparchive*. If you don't provide an extension, and the filename as entered doesn't exist, UNZIP adds ".zip". If you don't specify an operation, UNZIP will default to Extract. If you don't specify any arguments, UNZIP will display its command dialog.

UNZIP supports wildcards for the zip archive name and for the filenames to extract. UNZIP will prompt

before overwriting existing files. Your options at the prompt are explained in detail under [Page and File Prompts](#).

*path* specifies the path where files will be extracted. If *path* is not specified, files are extracted to the current directory.

UNZIP will automatically use the Zip64 extensions if the archive is in Zip64 format.

UNZIP sets two internal variables:

|                |                               |
|----------------|-------------------------------|
| %_unzip_files  | The number of files extracted |
| %_unzip_errors | The number of errors          |

**Option:**

- /= Display the UNZIP command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.
- /A:... Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with @file lists. See [@file lists](#) for details.

You can specify /A:= to display a dialog to help you set individual attributes.
- /C Display (on standard output) the contents of a file in the zip archive.
- /CRC Display the file CRCs (must be used with /V).
- /D Recreate the directory structure saved in the zip file.
- /E Extract the specified file(s). (This is the default.)
- /F Extract only those files that currently exist in the target folder, and which are older than the file in the zip archive.
- /I Save the "File Comment" (if any) for each extracted file to the NTFS description or the DESCRIPT.ION file.
- /Nt Don't update the CD / CDD [extended directory search](#) database (*JPSTREE.IDX*).
- /O Overwrite existing files. UNZIP normally prompts before overwriting an existing file; /O will suppress the prompt.
- /P Display the progress (0 - 100%) for each file as it is extracted.
- /Q Don't display filenames as they are extracted.
- /S Use the specified password to extract the file(s) from an encrypted archive. If you don't provide a password on the command line, UNZIP will prompt you to enter one.
- /TEST Test the integrity of the ZIP file (header and contents). Any errors will be displayed on STDERR.

- /U Extract files which either don't exist in the target folder, or which are older than the file in the zip archive.
- /V View the list of files in the archive (date, time, size, compression ratio, and filename). If the zip file is password protected, UNZIP will append a \* after the filename.

### 4.3.170 USBMONITOR

**Purpose:** Monitor USB device connection and disconnection

**Format:** USBMONITOR [/C [name]]  
USBMONITOR name CONNECTED | DISCONNECTED n command

**name** Device name  
**n** Number of repetitions (or **FOREVER**)  
**command** Command to execute when condition is triggered

[/C\(lear\)](#)

**Usage:**

The USB device name can include wildcards. You can use either the device ID or the "friendly" name for the device.

The command line will be parsed and expanded before USBMONITOR is executed, so if you want to pass redirection characters or variables to **command** you will need to protect them (by enclosing in single back quotes, doubling the %'s, or using command grouping).

If the last argument on the line is a single (, it is interpreted as the beginning of a command group. USBMONITOR will append the following lines (in a batch file) or prompt you for more input (at the command line) until it gets a closing ).

If you don't enter any arguments, USBMONITOR will display the USB devices it is currently monitoring.

The monitoring runs asynchronously in a separate thread. When the condition is triggered, the command will be executed immediately. This may cause problems if you try to write to the display or access files while the main **TCC** thread is also performing I/O. You may need to use [START](#) or [DETACH](#) in **command** to avoid conflicts.

USBMONITOR creates two environment variables when a device is connected or disconnected that can be queried by **command**. The variables are deleted after **command** is executed.

**\_usbdeviceid** The device ID (this will usually have special characters like & in the name, so you will probably need to use double quotes around the variable name to prevent **TCC** from parsing the special characters)

**\_usbname** The "friendly" name of the device

There is another variable that is updated after each trigger:

**\_usbcount** The number of times the command has been triggered

**Options:**

- /C** If **name** is specified, remove the monitor for that USB device. Otherwise, remove all USB monitors.

#### 4.3.171 VBEEP

**Purpose:** Flash the screen and beep the speaker

**Format:** VBEEP [frequency duration ...] [asterisk | exclamation | hand | question | ok]

|                    |                                                  |
|--------------------|--------------------------------------------------|
| <b>frequency</b>   | The beep frequency in Hertz (cycles per second). |
| <b>duration</b>    | The beep length in 1/18th second intervals.      |
| <b>asterisk</b>    | Plays the system default "asterisk" sound.       |
| <b>exclamation</b> | Plays the system default "exclamation" sound.    |
| <b>hand</b>        | Plays the system default "hand" sound.           |
| <b>question</b>    | Plays the system default "question" sound.       |
| <b>ok</b>          | Plays the system default "ok" sound.             |

See also: the [Length](#) and [Frequency](#) configuration options.

**Usage:**

VBEEP flashes the screen (by setting all attributes to their inverse), and generates a sound through your computer's speaker. You can use it in batch files to signal that an operation has been completed, or that the computer needs attention.

64-bit versions of Windows do not support the frequency/duration syntax of VBEEP.

You can include as many frequency and duration pairs as you wish. No sound will be generated for frequencies less than 20 Hz.. The default value for *frequency* is 440 Hz; the default value for *duration* is 2.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

#### 4.3.172 VER

**Purpose:** Display the **TCC** and operating system versions

**Format:** VER [/R]

[/R\(revision\)](#)

**Usage:**

Version numbers consist of a one or two-digit major version number, a separator, and a one- or two-digit minor version number. VER uses the default decimal separator defined by the current country information. The VER command displays version numbers for both **TCC** and Windows:

```
[c:\] ver
Take Command 16.0.25 Windows 7 [Version 6.1.7600]
```

**Option:**

**/R** Display the **TCC** and operating system internal revision level (if any), plus your registered

name.

### 4.3.173 VERIFY

**Purpose:** Enable or disable disk write verification or display the verification state

**Format:** VERIFY [ON | OFF]

**Usage:**

Disk write verification cannot actually be enabled under Windows. **TCC** supports VERIFY as a "do-nothing" command, for compatibility with CMD. This avoids **unknown command** errors in old batch files which use the VERIFY command. You can set verification for file copying with the [COPY /V](#) option.

If used without any parameters, VERIFY will display the state of the verify flag:

```
[c:\] verify
VERIFY is ON
```

### 4.3.174 VIEW

**Purpose:** Display the contents of files

**Format:** VIEW [/A:[[-+]rhsadecijopt /A /B /E /FIX /FLAT / GB /H /L /L:n ?LEN:n /O:xx /P /R /S:xx /T /TEXT /VH /W] file ...]

**file** One or more files.

|                                        |                                              |
|----------------------------------------|----------------------------------------------|
| <a href="#">/A: (Attribute select)</a> | <a href="#">/LEN:n (wrap length)</a>         |
| <a href="#">/A(SCII mode)</a>          | <a href="#">/O:xx (start at offset)</a>      |
| <a href="#">/B (EBCDIC)</a>            | <a href="#">/P(rint)</a>                     |
| <a href="#">/E (start at end)</a>      | <a href="#">/R (browse last directory)</a>   |
| <a href="#">/FIX</a>                   | <a href="#">/S:xx (search)</a>               |
| <a href="#">/FLAT</a>                  | <a href="#">/T (file tail)</a>               |
| <a href="#">/GB (greenbar)</a>         | <a href="#">/TEXT</a>                        |
| <a href="#">/H(ex)</a>                 | <a href="#">/VH (vertical hex)</a>           |
| <a href="#">/L(ast file)</a>           | <a href="#">/W (Take Command tab window)</a> |
| <a href="#">/L:n (start at line n)</a> |                                              |

See also [LIST](#).

#### File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

**Usage:**

VIEW provides a fast and flexible way to view a file, without the overhead of loading and using a text editor. VIEW is a replacement for the old [LIST](#) command.

For example, to display a file called *MEMO.TXT*:

```
view memo.txt
```

**Note:** VIEW is primarily intended for displaying the contents of ASCII, Unicode, and EBCDIC text files (i.e. alphanumeric characters arranged in lines separated by CR/LF). It can be used for other files which contain non-alphabetic characters or unusual line separators, but you may need to use hexadecimal mode (see below) to display or search these files.

If you don't specify any filenames on the VIEW command line, VIEW will first check to see if standard input has been redirected (i.e., with a pipe like "DIR | VIEW"). If so, VIEW will display the contents of standard input. If not, VIEW will display a dialog to allow you to select files to display.

VIEW can view CSV files as tables. CSV files are typically used to represent tabular data, where each line in the file represents a row of a table. Each line contains the text of each column in the row, separated by a comma (although other characters can be used, such as a tab). By default, VIEW will automatically recognize CSV files and will display them as a table - where all the columns have the same width (much like a spreadsheet). Although unlike a spreadsheet, the column widths in V are fixed (determined by the longest entry in the column) and cannot be resized. You can press the arrow button next to the new CSV Mode button in the toolbar to customize the CSV behavior. Press the CSV Mode button to toggle between CSV mode and standard text mode.

#### **Line Wrapping:**

When a line is too long to fit in the view, horizontal scroll bars appear at the bottom of the view, allowing you to scroll through the entire line. The horizontal scroll bars will appear when at least one line of the file being viewed is wider than the width of the view.

Sometimes scrolling through lines is not very convenient, particularly if a file contains many really long lines. In this case, lines can be wrapped. Lines may be wrapped in several ways:

##### **Wrap to Screen**

The lines are wrapped so that all text fits inside the file view. In this case, the horizontal scroll bars disappear. Screen wrapping may be toggled by selecting the Wrap to Screen command from the View menu, pressing the Wrap to Screen button on the toolbar, or by pressing Alt-W. Wrapping text to the screen may be permanently enabled by setting the Wrap lines to screen option in the Preferences Dialog box.

##### **Wrap to Length**

In this case, the lines are wrapped whenever they reach the wrap length. Wrapping may be toggled by selecting the Wrap to Length command from the View menu, pressing the Wrap to Length button on the toolbar, or by pressing Alt-L. The wrap length may be quickly changed by selecting the Set Wrap Length command from the View menu, or by pressing Ctrl-W. You may enter a new wrap length, or select a previously used length from the list.

##### **Wrap on Word Boundary**

Normally, lines will be wrapped at the exact position where they exceed the width of the view (if wrapping to screen), or the wrap length - even if it happens to be in the middle of a word. To ensure that lines are not split mid-word, select the Wrap on Word Boundary option from the Preferences Dialog box.

#### **Hex Mode:**

Hex mode displays a file as a series of hexadecimal (base 16) numbers together with the corresponding

ASCII character equivalent (this is also known as Debug Format).

The first 8 digits on each line represents a hex address which indicates the position (or offset) of the corresponding line in the file. This is followed by up to 16 hex numbers (or bytes) which correspond to the file data. The right hand side of the view consists of the ASCII character representation of the corresponding file data. If the hex byte does not correspond to a printable ASCII character, it is displayed as a "." (dot).

#### **Split Windows:**

The File window can be split in two by clicking on the Split Screen button on the toolbar and selecting Horizontal or Vertical Split Mode. It can also be split from the Split submenu of the View menu, or by pressing Alt+S. This allows you to view different parts of the same file in different windows.

Note that both windows must use the same display mode. For example, you cannot have one window in hex mode and the other in text mode. Also, if you enable line wrapping, the wrapping will apply to both windows.

#### **The Ruler:**

The ruler makes it easy to determine the position of a particular character. The ruler is displayed at the top of the File View and its format depends on the mode of the view.

If the view is in Text mode, the ruler consists of a sequence of incrementing numbers (starting at 1) which indicate the column number of the character below. In this case, the length of the ruler is determined by the length of the longest line in the file.

If the view is in Hex mode, the ruler always consists of 16 hex offsets (from 00 to 0F) which indicate the offset from the start of the line of the corresponding hex bytes displayed below.

The ruler may be dragged over any part of the file. The floating ruler may be removed by either double-clicking on it or by dragging it back to the top.

The ruler numbering usually starts at 1. To start from 0, right-click on the ruler with the Control key pressed and select the required option. Alternatively, you can press Ctrl+Alt+R to toggle the starting column.

Up to 10 floating rulers may be displayed while viewing a file. To create a new ruler simply drag it from the top (fixed) ruler. To close a ruler, double click on it. To close all rulers (but to remember their position), double click on the fixed ruler. To redisplay the floating rulers in their last position, double click again on the fixed ruler.

Right-clicking (or shift-clicking) on the ruler will cause a vertical grid line to be drawn at the clicked column position. The grid line will disappear once the button is released.

#### **Options:**

**/=**      Display the VIEW command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.

**/A:**      Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with [@file lists](#). See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes.

- /A** View the file in ASCII mode. This is the default mode and will only need to be specified in order to override an existing EBCDIC mode.
- /B** View the file in EBCDIC mode. VIEW normally automatically determines if a file is EBCDIC and automatically sets this mode.
- /E** Start viewing the file from the end instead of the beginning.
- /FIX:n** When viewing a file, the display may be fixed at a certain column position so that any text to the left of the fixed column will always be visible (ie, it will not scroll off the screen).
- /FLAT** Enables Flat Text Mode. This is a cross between text and hex modes. The file is displayed as text, however, control characters like line feeds and tabs are not expanded, and the file is always wrapped at the specified wrap length.
- /GB** Enables Greenbar Mode. Each alternating line is in a different color.
- /H** View the file in Hex mode.
- /L** Display the last file that was viewed. (This will be the first file in the Recent Files list.)
- /L:n** Start displaying the file from line number *n*. A solid blue line will appear at the top of the file, indicating that a non-zero start offset is being used.
- /LEN:n** Set the wrap length to *n*.
- /O:xx** Start displaying the file from offset *xx*.
- /P** Print the file and exit VIEW when finished.
- /R** When started with no parameters, VIEW will browse the current directory. By specifying the /R option, VIEW will display the directory that it last browsed.
- /S** The /S option is used to tell VIEW to start displaying the file at the position of a string match. The format of the /S command line option is as follows:

**/S:SearchString /SO:[CWRHUB] /SN:n /SC:Columns**

where SO can contain a series of letters which correspond to the options in the search dialog box. These can be one of:

- C** Match case
- W** Word Only
- R** Regular Expression
- H** Hex/Binary
- U** Unicode

**B** Search backwards (from end of file)

**SN** indicates which occurrence of the string to find. By default, the first match is found (n=1).

**SC** can be used to restrict the search to certain columns.

If the search string contains spaces, you must enclose it in double quotes.

- /T** Enable File Tailing. If data is added to the file while you are viewing it, it will automatically be updated. There is no need to press the Refresh button to see any changes since the file was loaded. This is particularly useful when viewing log files while they are still being updated.
- /TEXT** Open the files in text mode (opposite of /H). (This is the default.)
- /VH** Display the file in Vertical Hex Mode. This is a cross between Text and Hex modes. The file is displayed one line at a time (just as in text mode). However, each line is followed by 2 lines containing the hex code of each character in the line.
- /W** Display the VIEW window in a Take Command tab window.

#### 4.3.174.1 VIEW Command Line Options

Command line options may be entered on the **VIEW** command line.

For example,

**VIEW FileName [options]**

The following options are valid:

**/A** View the file in ASCII mode. This is the default mode and will only need to be specified in order to override an existing EBCDIC mode.

**/B** View the file in [EBCDIC mode](#).

**/E** **V** will start viewing the file from the **e**nd instead of the beginning.

**/** Enables [Flat Text Mode](#)

**/GB** Enables [Greenbar Mode](#)

**/H** This will force the file to be viewed in [Hex mode](#).

**/I** A new instance of **V** will be started (multiple calls to VIEW are usually handled by a single instance of **V**)

**/** Enables the [Ignore Consecutive CRs option](#)

**/IFF**Enables the [Ignore Form Feed option](#)

/L V will display the **last** file that it viewed. This will be the first file in the Recent Files list.  
/ V will start displaying the file from **line number nn**

/ Sets the wrap length to nn

/ V will start displaying the file from **offset xxxx**

/P Indicates that the specified file(s) are to be **printed**. Click here for [further printing options](#).  
/R When started with no parameters, V will browse the current directory. By specifying the /R option, V will display the directory that it last browsed. Note that this option is automatically added to all shortcuts that V creates for itself.  
/T Enable [File Tailing](#)  
/ This forces a file to be opened in [text mode](#) (opposite of /H)

/ See [Font Options](#)

VFON

T

/VH The file will be viewed in [Vertical Hex Mode](#)

/OEM V will display the file using the [DOS/OEM character set](#) (if available).

/ V will use the DOS/OEM character set when printing.

/ V will use the default character set.

Further command line options are explained in the following sections:

[/Delete Option](#)  
[/S Option](#)  
[/Fix Option](#)  
[Font Options](#)  
[Printing Options](#)  
[Text Only Printing Options](#)  
[EBCDIC Options](#)

### Notes

The options may also appear **before** the file name(s)

The options are **not** case sensitive.

You can use an equals (=) instead of a colon (:). For example, /L=20 instead of /L:20

#### 4.3.174.1.1 /Delete Option

The **/delete** option is used to tell **V** to delete the file once it has finished viewing it.

This option may be necessary if you are using **V** as an external file viewer. When using external file viewers, programs usually create temporary files and then launch the file viewer to display the temporary file. It is up to the program that launches the file viewer to delete any temporary files it has created.

However, not all programs are well behaved and it is sometimes not possible to delete the file because **V** may still have the file open. If you know that the program in question always creates temporary files before it passes them to **V**, you should use the **/delete** flag.

**\*\*\* Please use this option with caution \*\*\***

#### 4.3.174.1.2 Find String Option: /S

The **/S** option is used to tell **V** to start displaying the file at the position of a string match.

The format of the **/S** command line option is as follows:

**V** **Filename** **/S: searchString /SO:[CWRHUB] /SN:n /SC:Columns**

where

**SO** can contain a series of letters which correspond to the options in the [search dialog box](#). These can be one of:

**C** Match case  
**W** Word Only  
**R** Regular Expression

- H** Hex/Binary
- U** Unicode
- B** Search backwards (from end of file)

**SN** indicates which occurrence of the string to find. By default, the first match is found (n=1).

**SC** can be used to [restrict the search to certain columns](#).

If the search string contains spaces, you must enclose it in double quotes.

## Examples

Start viewing at the second occurrence of "Error"

**V** Filename /S:Error /SN:2

Start viewing at the last line that contains the \*word\* "Error"

**V** Filename /S:Error /SO:WB

Note that the first example will match "errors", whereas the second will not.

Start viewing at the first line that \*begins\* with "Error Log"

**V** Filename /S:"^Error Log" /SO:R

### Note

The /S parameter will not work on wild cards - you must specify a valid file name.

For example, you cannot specify "**V \*.txt /S:Error**" hoping that **V** will display the first txt file that contains "Error"

#### 4.3.174.1.3 /Fix Option

The **/FIX** option tells **V** to [fix the columns](#) when viewing the specified file(s).

The column position may also be specified on the command line.

For example,

```
V TABLE.TXT /FIX:6
```

If no column position is specified, the most recent fixed column position is used.

To disable fixed columns, specify a column position of zero. That is,

```
V TABLE.TXT /FIX:0
```

#### 4.3.174.1.4 Font Options

You may specify the display font on the command line by using the **VFONT** option as follows:

```
/VFont:"Font Name",size
```

For example:

```
V Filename /VFont:"Courier New",10
```

#### Notes

The **size** option is optional. If not specified, it will default to 10 point.

The **/FONT** option is used to specify what font will be used when [printing](#).

#### 4.3.174.1.5 Viewing Redirected Output

When using a Command Prompt, it is awkward to view the output of a command (like DIR or GREP) when that command outputs more lines than can fit on the screen.

Sometimes, the Command Prompt allows you to scroll back through the output. However, you still can't search, or select and copy the output.

Typically, the **more** command is used as a filter to pause the output a screen at a time. However, it is still very limited - you can't scroll back or search.

You can solve all the above problems by using **V** to view the redirected output. You will be able to scroll and search the output, and even [save it to a file](#).

No option is required to view redirected output. If you do not provide a file name on the command line, **V** will automatically check if there is any redirected output.

#### 4.3.174.1.6 Printing Options

The **/P** option is used on the command line to print the specified file(s). **V** will immediately start printing the file (in a minimized state), and will exit when it has finished. If you have **V** installed in the tray, a separate instance of **V** will be launched to do the printing.

The Print Dialog box will not appear when printing from the command line (unless the **/PD** option is specified). All the current print settings will be used for printing. Most of these settings can be overridden by specifying them on the command line. The following command line options are available:

**/Portrait**  
**/Landscape**  
**/2up**

If one of the above 3 options is used, the **/P** option is implied and does not need to be specified.

|                                |                               |
|--------------------------------|-------------------------------|
| <b>/Font:"Font Name",size</b>  | eg, /Font:"Courier New",10    |
| <b>/Printer:"Printer Name"</b> | eg, Printer:"HP LaserJet III" |
| <b>/AM:0.5</b>                 | Set ALL margins to 0.5 inches |
| <b>/LM:0.5</b>                 | Left margin                   |
| <b>/RM:0.5</b>                 | Right                         |
| <b>/TM:0.5</b>                 | Top                           |
| <b>/BM:0.5</b>                 | Bottom                        |
| <b>/Header:"Header Text"</b>   | /Header:"%f;Page %p"          |
| <b>/Footer:"Footer Text"</b>   |                               |
| <b>/Copies:2</b>               |                               |
| <b>/From:2 /To:3</b>           | Print From Page 2 to Page 3   |
| <b>/L1:1 /L2:200</b>           | Print From Line 1 to Line 200 |
| <b>/Pagelen:60</b>             | Override Page Length          |

|                                |                                                                          |
|--------------------------------|--------------------------------------------------------------------------|
| <b>/Wrap</b>                   | Wrap Long Lines                                                          |
| <b>/Wrappage</b>               | Wraps long lines onto a new page instead of onto a<br>new line           |
| <b>/EOL</b>                    | Print End of Line                                                        |
| <b>/PLN</b>                    | Print Line Numbers                                                       |
| <b>/IFFP</b>                   | Ignore Form Feeds                                                        |
| <b>/ODD</b>                    | Only print odd numbered pages                                            |
| <b>/EVEN</b>                   | Only print even numbered pages                                           |
| <b>/PX</b>                     | Use <a href="#">Text Only printing</a>                                   |
| <b>/PB</b>                     | Use <a href="#">Raw/Binary printing</a>                                  |
| <b>/PGB</b>                    | Enables <a href="#">Greenbar</a> printing                                |
| <b>/Profile:"Profile Name"</b> | Use the settings stored in the specified <a href="#">printer profile</a> |

You may also specify the **/PD** option which will cause the Print Dialog box to be displayed, allowing the user to override any print options.

If your printer supports duplexing (double sided printing), you may also specify one of the following options:

|              |                                          |
|--------------|------------------------------------------|
| <b>/dups</b> | Enable duplexing with short edge binding |
| <b>/dupl</b> | Enable duplexing with long edge binding  |

If you do not specify a print setting, the current (most recent) setting will be used. To disable a setting, prefix the corresponding option with **NO**.

For example, if the default setting was to print a Header, you would have to specify the **/NOHEADER** option to disable the header. You can also use, **NOFOOTER**, **NO2UP**, **NOWRAP**, **NOEOL**, **NOPLN** and **NOPAGELEN**.

#### **/NOP option**

When the **/P** option is used (or implied), **V** will immediately start to print the file. However, if you would like the user to be able to view the file before it is printed, you need to specify the **/NOP** option. **V** will retain the command line parameters and use them when the user decides to print the file.

#### **/PAGELIMIT:xxx option**

The **/PAGELIMIT** option is used to specify the maximum number of pages that can be printed. This can be used to prevent users from accidentally printing the entire contents of very large files.

#### **Notes**

Case is not important (ie, **/Font** or **/font** can be used).

Make sure that there is no space before or after the ":" in each option, and no space before or after the comma separating the font name and font size.

The header text, printer name and font name should be enclosed in "quotes".

If not specified, the default margins will be used. However, if one margin is specified on the command line, they must ALL be specified. Any margin not specified will default to ZERO.

## 4.3.174.1.7 Text Only Printing Options

The Text Only Printing options can be entered on the command line.

They correspond to the options in the [Text Only dialog box](#), and are as follows:

|                         |                                                    |
|-------------------------|----------------------------------------------------|
| <code>/TOAM:nn</code>   | Set *all* margins to nn                            |
| <code>/TOLM:nn</code>   |                                                    |
| <code>/TORM:nn</code>   |                                                    |
| <code>/TOTM:nn</code>   |                                                    |
| <code>/TOBM:nn</code>   | Set left, right, top and bottom margins            |
| <code>/TOW:nn</code>    | Set page width to nn                               |
| <code>/TOH:nn</code>    | Set page height to nn (ie, page length)            |
| <code>/TOLD:n</code>    | Set the Line Delay to n                            |
| <code>/TOPD:n</code>    | Set the Page Delay to n                            |
| <code>/TODISABLE</code> | To disable Text Only printing                      |
| <code>/TOSOF:xxx</code> | String to send before the start of file is printed |
| <code>/TOEOF:xxx</code> | String to send at end of file                      |
| <code>/TOEOL:xxx</code> | String to send after each line                     |
| <code>/TOEOP:xxx</code> | String to send after each page                     |

To include control characters in a string, you must specify their 2 character hex code prefixed by a "%". For example to send a CR/LF at the end of each line and a FF (form feed) at the end of each page, you would specify the following options:

[/TOEOL:%0d%0a /TOEOP:%0c](#)

#### 4.3.174.1.8 EBCDIC Options

You can specify the [EBCDIC](#) record format and record length by using the **RECFM** and **LRECL** options.

##### **/RECFM=xxx**

Use this to specify the record format. The record format can be one of **V**, **VB**, **F** or **U**. If the file uses [carriage control](#), you can also specify the type of carriage control by appending one of **A**, **M** or **Z**. An **S** may be appended to signify that the file contains ASCII characters (instead of EBCDIC).

##### **/LRECL=nn**

Use this to specify the record length for fixed length files (RECFM=F). This is not required for RECFM=V/VB/U.

#### Examples

- ▼ Filename /RECFM=F /LRECL=132
- ▼ Filename /RECFM=FM /LRECL=80
- ▼ Filename /RECFM=VBA
- ▼ Filename /RECFM=VS

[Click here for further details on EBCDIC record formats](#)

#### Note

**V** can usually auto-detect RECFM=V/VB files, so it is not necessary to specify these formats on the command line. However, **V** cannot detect if the file contains carriage control. If it does, you will need to use the RECFM option to specify the type of carriage control (eg, /RECFM=VA).

#### 4.3.174.1.9 Sending Error Reports

If **V** crashes, it will display a message saying that it encountered an *unexpected problem* and that it cannot continue.

Press the *Send* button to send a detailed report to [fileviewer.com](#) for analysis.

Error reports can also be sent to [fileviewer.com](#) by selecting *Send Error Report* from the Help menu

You will be presented with a list of crash files (**Crash-xxxx.dat**) and a log file (**V.log**). Simply select the files that you want to send and press the *Send* button.

Please include a description of what you were doing just before the crash.

#### 4.3.174.2 The File View

The File View is the view in which the file is displayed. The view can be in one of two modes - [Text](#) and [Hex](#), and can be configured in several ways.

The font in which the file is displayed may be selected from the [Fonts](#) section of the [Preferences](#) Dialog box. Note that V only supports non-proportional (or fixed pitch) display fonts - like Courier.

Right-clicking on any part of the File View will display a pop-up menu containing most of the available commands.

The following sections describe the File View in greater detail:

[The Toolbar](#)  
[Text Mode](#)  
[Hex Mode](#)  
[Unicode Files](#)  
[Flat Text Mode](#)  
[End of Line](#)  
[Tabs](#)  
[Start Offset](#)  
[EBCDIC](#)  
[The Ruler](#)  
[Line Numbers](#)  
[Line Wrapping](#)  
[Column Fixing](#)  
[Line Lengths](#)  
[OEM Character Set](#)  
[Bookmarks](#)  
[Scrolling](#)  
[Searching](#)  
[Goto](#)  
[Block Marking](#)  
[File Chunks](#)  
[Greenbar Mode](#)  
[File Tailing](#)

[Click here for details on how to configure V to view multiple files.](#)

#### 4.3.174.2.1 The Toolbar

Below is the default toolbar. It can be customized so that the buttons are re-arranged or deleted (right-click on the toolbar and select Customize).



##### Directory

Not available.

##### Refresh

Refreshes (or reloads) the file. It is possible that the file being viewed is also being modified by another program (eg, a log file). In order to be able to view any data appended to the file since the file was opened, you need to refresh it (unless [File Tailing](#) is enabled).

##### Previous/Next

Displays the previous/next file in the file list. If you select several files in the Directory View, the file list will consist just of the files selected, otherwise the file list will consist of all files in the directory. If you position the mouse over the Previous/Next buttons, the name of the corresponding file will be displayed. Note that these buttons work differently from the Back/Forward buttons in a browser.

##### File List

This brings up a list box containing all the files in the current file list. As above, if you have selected several files in the Directory View, this list will only contain the selected files. Otherwise, it will contain all the file in the current directory. To view another file, simply select it from the file list. Note that this feature is only available once V has been registered.

##### Find

[Searches for a string.](#)

##### Find Next/Find Previous

Searches for the next (or previous) occurrence of a string.

##### Goto

Goes to a specified position in the file. [Click here for further details.](#)

##### Clipboard

Copies (or appends) selected text to the clipboard. [Click here for further details.](#)

##### Text Mode

Displays the file in [Text Mode](#)

##### Hex Mode

Displays the file in [Hex Mode](#)

##### Wrap To Screen/Wrap To length

[Click here for further details on line wrapping.](#)

 **Line Numbers**

Toggles [line numbers](#) on/off in Text mode.

 **EOL**

This specifies whether an **End Of Line** character will be displayed at the end of every line. This is useful when viewing files with trailing spaces. Note that this option is not available in Hex mode. The character used as the End Of Line terminator may be specified in the [Fonts](#) tab of the Preferences dialog box. The **EOL** mode can also be toggled by pressing the **Enter** key. Note that the **EOL** character will not be displayed at the end of a line that has been wrapped. It will only be displayed if it corresponds to an actual end of line character in the file.

 **Tail**

This enables [File Tailing](#).

 **Greenbar**

Clicking on this button enables/disables [Greenbar Mode](#). Clicking on the arrow portion of the button allows you to modify the [Greenbar Options](#).

 **Tools**

Clicking the Tools button displays a further menu where you may select one of the following:

 **MD5/CRC32**

This calculates the MD5 and CRC32 of the file being viewed.

 **Word/Line Count**

This counts the number of words and lines in the file.

 **Zoom**

You may easily increase (or decrease) the size of the current font by using the + and - keys. This does not do a "bitmap zoom", but simply selects the next point size in the current font. If a larger (or smaller) point size is not available, nothing will happen. Note that this will not work properly with the Terminal font.

 **Send**

This will construct an email message using your email client and will attach the file being viewed. Note that this will only work if MAPI (or Windows Messaging) is installed on your system.

#### 4.3.174.2.2 Text Mode

The Text mode displays the contents of a file exactly as they are stored on disk - much like an editor does.

The Text mode may be enhanced by adding [line numbers](#), adding a [ruler](#) and [wrapping lines](#). These topics are discussed further in later sections.

When **V** opens a file, it determines whether it is a text or binary file, and displays the file in Text or [Hex](#) mode respectively. Basically, text files contain only alphanumeric, punctuation and [new line](#) characters. If files contain characters other than these, they will be displayed in Hex mode.

It takes a little bit longer for files to be displayed in Text mode rather than Hex mode. Files that are loaded in Hex mode will display almost instantaneously - regardless of size (1 byte or 500Mb). However, displaying files in Text mode is different. To display the file properly (and to handle the scroll bars correctly), **V** needs to know the number of lines in the file, and also the length of the longest line. As files get larger, it naturally takes longer to do this. Normally, you will not notice any delay unless the files are at least 8Mb in size.

#### **File Chunks**

To enable fast loading of even **very large files** (hundreds of Mb to several Gb), **V** can view files in **chunks**, instead of loading the entire file. [Click here for further details on file chunks](#).

#### **File Tailing**

If you want **V** to automatically refresh a file as it is being viewed, you need to enable File Tailing. [Click here for further details](#).

#### **Notes**

Binary files (like ZIP and EXE files) can be viewed in Text mode, although it usually doesn't make sense to do so. If such files are viewed in text mode, many strange characters will be displayed. These strange characters correspond to non-printable (or control) characters and will differ depending on which font is selected.

Sometimes **V** can incorrectly decide that a text file is a binary file, and display it in Hex mode. This usually happens if a file contains an unexpected control character (eg, line drawing characters). In this case, just click on the Text icon on the toolbar (or press **Alt-H**) to display the file in Text mode. If you find that **V** is incorrectly displaying most of your text files in Hex mode, you can force **V** to always view them in text mode by enabling the [Always Open as Text](#) option in the [Preferences](#) Dialog box. However, by enabling this option, even ZIP and EXE files will initially be displayed as text.

#### 4.3.174.2.3 Unicode Files

Most text files are stored using ASCII characters - each character is encoded using one byte (8 bits). This means that we can have a maximum of 256 different characters. This isn't a problem in most English speaking environments, but it does become a problem once you start encoding characters in different languages.

Unicode is a standard for encoding characters that tries to address the problem of encoding all possible international characters into a single, unified format.

As with most standards, there are several flavors to choose from. **V** supports UCS-2 and UTF-8. (See the note below regarding UTF-16)

#### Status Bar Indicator

**V** will automatically detect most Unicode files and display them accordingly, including files with foreign characters. **UNI** will be displayed in the status bar to indicate that the file is a Unicode file. **ANS** (for ANSI) will be displayed in the status bar when the file is not a Unicode file.

If **V** does not guess the correct encoding, you can click on the **UNI/ANS** indicator in the status bar and select the correct encoding (assuming that you know what it is).

#### Font Substitution

**V** does not support font substitution (or font fallback). Under font substitution, if the selected font does not contain a particular character, the program will try to use another font to display that character. Since **V** does not do font substitution, it is very important to use a font that contains all the characters to be displayed. In particular, the standard **Courier** font should not be used to display Unicode files - **Courier New** should be used instead.

#### UCS-2 vs UTF-16

Strictly speaking, **V** does not fully support UTF-16 - it only supports UCS-2 (which is the outdated predecessor to UTF-16).

UCS-2 is a fixed length encoding that encodes all characters to a 16 bit value (from 0 to FFFF). UTF-16 is a variable length encoding capable of encoding the entire Unicode range of characters. In particular, UTF-16 can be used to encode characters greater than FFFF.

However, in most cases, UCS-2 and UTF-16 are identical. If users encounter any problems viewing Unicode files, please contact [v@fileviewer.com](mailto:v@fileviewer.com) (preferably attaching a copy of the Unicode file).

#### **Notes**

**V** does not support UTF-32

**V** does not support RTL (Right To Left) display

#### 4.3.174.2.4 Flat Text mode

This is a cross between [text](#) and [hex](#) modes.

The file is displayed as text, however, control characters like line feeds and tabs are not expanded, and the file is always wrapped at the specified wrap length.

This is useful for displaying files of fixed length records, where records may contain control characters (eg, packed decimal fields).

See the [Wrap Here](#) section for details on how to quickly wrap lines at different line lengths.

#### 4.3.174.2.5 End of Line (EOL)

When displaying files in text mode, **V** will automatically start a new line whenever a line terminator is encountered (unless the file is displayed in [Flat Text](#) or [Wrap to Length](#) mode).

Most files created on Windows use a Carriage Return/Line Feed pair as a line terminator (CR/LF). That is, a Carriage Return followed by a Line Feed. However, some files use a single CR or a single LF as a line terminator. **V** will also start a new line whenever a single CR or LF is encountered.

The **End of Line** submenu (on the View menu) allows you to configure how you want **V** to handle line terminator characters.

##### **Display EOL Marker**

Enabling this will display the End Of Line marker at the end of every line. This is equivalent to pressing the [End of Line button on the toolbar](#). The character displayed as the EOL marker can be set in the [Fonts section of Preferences](#).

##### **Ignore Form Feeds**

Form feed characters (ASCII 12 or Ctrl-L) are generally used to signify a page break. **V** uses form feeds to start a new page when printing, and also treats a form feed as a line terminator (since a new page implies a new line).

If you do not want **V** to treat a form feed as a line terminator, you can enable this option. The form feed character will still be displayed, but a new line will not be started. Note that this option only applies to *viewing* files. If you also do not want **V** to start a new page when *printing*, you need to enable "[Ignore Form Feeds](#)" on the [Print dialog box](#).

##### **Customize EOL Options**

If you want to change any of the default EOL options, select *Customize EOL Options* from the *End Of Line* submenu.

Select [Use Default EOL Options](#) to revert to the default EOL behavior.

The following options are available:

##### **Ignore Single CR (without LF)**

This requires a CR to be following by a LF for it to be treated as a line terminator.

##### **Ignore Single LF (without CR)**

This requires a LF to be following by a CR for it to be treated as a line terminator.

##### **Ignore Consecutive CRs**

Some files have a strange EOL combination - CRCRLF. That is, 2 carriage returns followed by a line feed. Some users want this displayed as 2 lines, others as one. By default, **V** will treat this a 2 line terminators. If you enable this option, **V** will ignore the first CR and treat CRCRLF as a single line terminator.

**Use the following Custom EOL characters(s)**

Enable this option if you want to set custom EOL characters. Simply enter the characters in the space provided. **V** will start a new line whenever it encounters any of these characters. Multiple EOL characters are allowed, but *each* character will be treated as an EOL. That is, multiple characters will not be treated as a multiple character EOL combination.

**Disable default EOL characters**

When setting custom EOL characters, the default EOL characters (CR, LF) will still be treated as line terminators. Enable this option if you do not want CR and LF to be treated as line terminators.

#### 4.3.174.2.6 Tabs

When displaying text files, tabs will be expanded according to the number of characters specified in [Tab Size](#).

Specifying a Tab Size of 1 causes tabs to be treated as spaces.

Tabs may also be made visible by selecting "Tabs->Show Tabs" from the View menu. In this case, a tab will not be expanded and will be displayed according to the corresponding character in the selected font.

#### 4.3.174.2.7 Start Offset

**V** allows you to specify a non-zero **start of file** position - any data before this position will be ignored.

To set the start offset, right-click on the position in the file where you want to start viewing from and select **View/Layout->StartOffset->StartFromHere**

Alternatively, you can specify an absolute position by selecting **Set Offset** from the above right-click menu or from the main **View->StartOffset** menu.

A solid blue line will appear at the top of the file, indicating that a non-zero start offset is being used.

Note that the start offset is not "sticky". That is, it is reset to zero once a new file is viewed.

This is useful if you want view files with fixed line lengths, but the fixed lines do not begin from the start of the file.

#### 4.3.174.2.8 The Status Bar

The status bar of the File View usually consists of 3 panes:

##### [Pane 1 - Current Position](#)

Usually this will contain the current line number. If you press the left mouse button at any position and keep it pressed, the corresponding line number and column will be displayed. If you have enabled "[Hex offset in status bar](#)" (in the File Options tab of [Preferences](#)), the hex offset of the position will also be displayed.

In Hex mode, the corresponding file offset will be displayed. Also, as a block is being highlighted, the [start and end positions of the block are displayed](#), as well as the number of characters highlighted.

The offsets are displayed in both hex and decimal. For example,

**"Offset: 669h->8e8h (1641d->2280d) Len=640 (280h)"**

Note that as long as a block of text remains highlighted, its details will be displayed in this pane - even if the block has been scrolled out of view.

##### [Pane 2 - Position as Percentage](#)

This displays a number between 0 and 100 and represents the current line (ie, the last line in the view) as a percentage of the number of lines in the file.

If the file is [paginated](#), the current page number and the total number of pages will be displayed instead of the percentage.

##### [Pane 3 - File Details](#)

This displays the file size, the number of lines in the file and the date and time the file was last modified.

Unicode and EBCDIC

When **V** displays [Unicode](#) or [EBCDIC](#) files, **UNI** or **EBC** will be displayed in the status bar. You may click on this area of the status bar to display a menu of available options.

#### **Notes**

When the file is being displayed as [chunks](#), a fourth status pane appears which displays which chunk is currently being viewed.

Left-clicking on a character while viewing a file displays the hex offset of the character on the status bar (providing the Hex Offset option is enabled in Preferences).

## 4.3.174.2.9 EBCDIC Mode

EBCDIC stands for **E**xtended **B**inary **C**oded **DI**nterchange **C**ode, and is the character set used by most IBM mainframes to store documents (in preference to the more commonly used ASCII character set).

If you try to view an EBCDIC file with a standard ASCII file viewer/editor (eg, notepad), the text will appear as a stream of unprintable control characters. For example, the EBCDIC code for the number zero is **hex F0**, which is not a printable character in ASCII.

When **V** opens a file, it automatically tries to determine if it is ASCII or EBCDIC. If a file is EBCDIC, **EBC** will be displayed on the bottom status bar.

If an EBCDIC file is incorrectly displayed as ASCII, you can view it as EBCDIC by pressing Alt+B (or selecting EBCDIC from the View menu).

Once in EBCDIC mode, you may modify EBCDIC viewing options by selecting **EBCDIC Options** from the View menu, or by clicking on **EBC** in the status bar.

**V** views EBCDIC files by mapping each EBCDIC character to the ASCII equivalent before displaying. However, there are at least 6 incompatible versions of EBCDIC (all having non-contiguous letter sequences and missing punctuation characters). In order to support all of these mappings (and more), **V** defines a default mapping which can then be modified in the [EBCDIC](#) tab of the [Preferences](#).

**V** supports files with variable length records (RECFM=V) and fixed length records (RECFM=F).

[Click here for further details on viewing EBCDIC files.](#)

#### 4.3.174.2.10 The Ruler

The ruler makes it easy to determine the position of a particular character - you will no longer leave finger prints on your monitor as you count!

The ruler is displayed at the top of the File View and its format depends on the mode of the view.

#### Text Ruler

|           |           |           |           |           |
|-----------|-----------|-----------|-----------|-----------|
| <b>0</b>  | <b>10</b> | <b>20</b> | <b>30</b> | <b>40</b> |
| 123456789 | 123456789 | 123456789 | 123456789 | 123456789 |

If the view is in Text mode, the ruler consists of a sequence of incrementing numbers (starting at 1) which indicate the column number of the character below. In this case, the length of the ruler is determined by the length of the longest line in the file.

#### Hex Ruler

|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

If the view is in Hex mode, the ruler always consists of 16 hex offsets (from 00 to 0F) which indicate the offset from the start of the line of the corresponding hex bytes displayed below.

#### Floating the Ruler

The ruler may be dragged over any part of the file. The floating ruler may be removed by either double-clicking on it or by dragging it back to the top.

#### Starting Column

The ruler numbering usually starts at 1. To start from 0, right-click on the ruler with the Control key pressed and select the required option. Alternatively, you can press Ctrl+Alt+R to toggle the starting column.

#### Multiple Rulers

Up to 10 floating rulers may be displayed while viewing a file. To create a new ruler simply drag it from the top (fixed) ruler. To close a ruler, double click on it. To close all rulers (but to remember their position), double click on the fixed ruler. To redisplay the floating rulers in their last position, double click again on the fixed ruler.

#### Displaying Grid Lines

Right-clicking (or shift-clicking) on the ruler will cause a vertical grid line to be drawn at the clicked column position. The grid line will disappear once the button is released.

The color of the grid line can be customized in the [File View section](#) of *Customize Colors* (selected from the View menu).

#### **Notes**

- In Text mode, the ruler can be shifted by one character to the right (ie, the first column will be treated as column 0 instead of column 1) by setting the [Start Text Ruler at 0](#) option in the [Preferences](#) Dialog box.
- The ruler is always displayed in the same font as the file.

- The floating ruler and grid lines only become fully functional once V has been registered.

#### 4.3.174.2.11 Line Numbers

When viewing the file in Text mode, the corresponding number of the line can be displayed to the left of the line by enabling the **Line Numbers** option in the View menu. The line numbers display can also be toggled by pressing the line numbers icon in the [toolbar](#).

Look at the [Line Numbers Configuration](#) section for further details on how to customize the line number display.

#### Line Numbers in Chunks

If a file is being viewed in [chunks](#), **V** will display the correct line numbers as long as consecutive chunks are loaded. However, if you load a chunk without having viewed the previous chunk, **V** will start the line numbers from 1.

For example, if you load the last chunk in the file after having loaded the first chunk, the line numbers in the last chunk will start from 1. If you need to know the exact line number, you will have to load the entire file.

#### Notes

Line numbers always start at 1.

The line numbers are not **fixed**. That is, the line numbers are scrolled off the screen when the line is scrolled horizontally. You can fix the line numbers (so they do not scroll) by enabling *Fix Line Numbers* from the *Fix LHS* submenu of the *View* menu.

Although line numbers in Text mode are optional, the hex addresses in Hex mode are always displayed.

#### 4.3.174.2.12 Line Wrapping

When a line is too long to fit in the view, horizontal scroll bars appear at the bottom of the view, allowing you to scroll through the entire line. The horizontal scroll bars will appear when at least one line of the file being viewed is wider than the width of the view.

Sometimes scrolling through lines is not very convenient, particularly if a file contains many really long lines. In this case, lines can be wrapped. Lines may be wrapped in several ways:

##### Wrap to Screen

The lines are wrapped so that all text fits inside the file view. In this case, the horizontal scroll bars disappear. Screen wrapping may be toggled by selecting the **Wrap to Screen** command from the **View** menu, pressing the Wrap to Screen button on the toolbar, or by pressing **Alt-W**. Wrapping text to the screen may be permanently enabled by setting the [Wrap lines to screen](#) option in the [Preferences](#) Dialog box.

##### Wrap to Length

In this case, the lines are wrapped whenever they reach the **wrap length**. Wrapping may be toggled by selecting the **Wrap to Length** command from the **View** menu, pressing the Wrap to Length button on the toolbar, or by pressing **Alt-L**.

The wrap length may be quickly changed by selecting the **Set Wrap Length** command from the **View** menu, or by pressing **Ctrl-W**. You may enter a new wrap length, or select a previously used length from the list.

##### Wrap on Word Boundary

Normally, lines will be wrapped at the exact position where they exceed the width of the view (if wrapping to screen), or the wrap length - even if it happens to be in the middle of a word. To ensure that lines are not split mid-word, select the [Wrap on Word Boundary](#) option from the [Preferences](#) Dialog box.

##### Wrap Here

[Flat text mode](#) allows you to display the text file as fixed length records by specifying a record length (or wrap length).

Using the "Wrap Here" command, you can easily change the wrap length without actually entering a number.

All you do is right click on the position where you want the **first** record to end (or wrap) and then select "Wrap Here" from the Wrap menu.

##### **Note**

The **Wrap Lines to screen** option in the Preferences Dialog box, does not apply to printing. There is a separate option in the [Print](#) Dialog box which enables line wrapping when printing. However, the **Wrap on Word Boundary** option applies to both printing and display.

#### 4.3.174.2.13 Column Fixing

When viewing a file, the display may be fixed at a certain column position so that any text to the left of the fixed column will always be visible (ie, it will not scroll off the screen). To do this, select the **Fix Column** option from the View menu. Select **Set Fixed Column** to specify the column position.

You may also just fix the line numbers by selecting the **Fix Line Numbers** option from the View menu.

**Note:** Unregistered versions cannot fix the display past column 6.

#### 4.3.174.2.14 DOS/OEM Character Set

Most Windows applications use the **ANSI** character set to display text. The actual character displayed depends on the font selected.

Before Windows and GUI programs existed (in the DOS days), special line drawing characters were used to "draw" simple boxes and frames in standard text files. These special characters were a part of what was called the **OEM** or IBM character set.

Windows (True Type) fonts do not usually support these line drawing characters. These characters will usually be replaced by some strange symbol.

However, some fonts (like Courier New) support both ANSI and OEM character sets. By selecting the **DOS/OEM Char Set** option (from the View menu) the OEM character set will be selected (if available), and your lines and boxes will be drawn correctly.

Note that this will only work if the selected font supports the OEM character set. If it doesn't, this option will have no effect. For example, an OEM character set is available for Courier New but not for Courier.

#### 4.3.174.2.15 Display Fonts

The file is displayed using the currently selected **Screen Font** which is specified in the [Preferences](#).

If you find that you regularly use different fonts, you may create a fonts list and then select the required font from the Fonts menu.

To create or modify the fonts list, select **Fonts->Organize** from the View menu.

Organizing fonts is similar to [Organizing Favorites or User Commands](#).

Once a font has been added to the list, it may be selected from the **View->Fonts** menu.

A shortcut key can also be assigned to a font so that the font will always be selected whenever the corresponding shortcut key is pressed.

**Note:** V maintains different fonts for [text](#) and [hex](#) modes.

#### 4.3.174.2.16 Greenbar Mode

Files can be viewed (and printed) with a greenbar effect. This is where each alternating line is a different color. To enable, simply press the Greenbar icon in the toolbar.

##### [Greenbar Options](#)

The greenbar effect can be customized by clicking on the small arrow next to the Greenbar icon on the toolbar.

##### **Greenbar Background Color**

Click on this button to select the Greenbar color. Right-click on the button to select one of several pre-defined colors. You can select a different greenbar color for display and printing.

##### **Number of lines to greenbar**

By default, the greenbar color will alternate after every line. This option allows you to increase the number of consecutive lines that are "greenbarred".

##### **Start greenbar at line 1**

Enable this option if you want the greenbar coloring to start from line 1.

##### **Do not greenbar line numbers**

If you enable this option, the greenbar colors will not extend to the line number portion of the line.

##### **Do not change color for wrapped lines**

Enabling this option ensures that the greenbar color does not change if it has been wrapped. That is, the entire line will be displayed in the same color, regardless of how many lines it wraps to.

#### 4.3.174.2.17 File Attributes / Line Lengths

If the File Attributes command is selected in [text mode](#), the dialog box that displays the file attributes will contain an extra 4 lines of file details:

##### **Longest Line**

Displays the length (and number) of the longest line in the file.

##### **Shortest Line**

Displays the length (and number) of the shortest line in the file - including empty lines. That is, if the file contains an empty line, this length will be zero.

##### **Shortest non-empty**

Displays the length (and number) of the shortest non-empty line in the file. That is, this length will never be zero

##### **Line Terminator**

Displays the character (or character pair) used by the file to delimit lines. Depending on the origin of the file, the character(s) used to terminate lines are usually one of:

**CR** The terminator is a single **carriage return** character (hex 0D). This terminator is uncommon.

**LF** The terminator is a single **line feed** character (hex 0A). This is the standard terminator used on Unix systems.

**CR/LF** A carriage return followed by a line feed is used to terminate lines. This is the standard terminator used on PC based systems.

**LF/CR** As above, but the line feed is placed before the carriage return. This is very uncommon.

**CR/CR** Two carriage returns are used to terminate lines. This is also very uncommon.

### **Notes**

This extra information is not displayed if the file is being viewed in [hex mode](#), or if the File Attributes command has been selected from the Directory View.

If multiple lines share the same length, only the first line in the file with that length is displayed.

If a [file chunk](#) is being viewed, the line length information only applies to the lines in that chunk, and not the entire file.

The number of lines in the entire file (together with the total number of words) may be displayed by selecting *Word/Line Count* from the Tools menu.

#### 4.3.174.2.18 Hex Mode

Use this command to view a file in Hex format. This mode is generally used for binary (non-text) files, although text files can also be viewed in Hex mode.

Vertical Hex mode can be enabled by selecting **Text+Vertical Hex** from the View menu (or by pressing Alt+J). This is a cross between Text and Hex modes.

[Click here for further details on viewing files in hex \(and vertical hex\) mode.](#)

#### 4.3.174.2.19 File Tailing

**File Tailing** refers to the ability to automatically refresh a file as it is being modified.

By default, File Tailing is disabled. To enable it, press the Tail icon on the toolbar (or select Tail from the File menu).

If data is **added** to the file while you are viewing it, it will automatically be updated. There is no need to press the Refresh button to see any changes since the file was loaded.

This is particularly useful when viewing log files while they are still being updated.

Because there is some overhead in File Tailing, it is best not to enable it when it is not needed. To overcome the problem of the user accidentally leaving File Tailing enabled, **V** always disables File Tailing on startup.

If you want **V** to restore its previous Tailing state on startup, enable "Save File Tailing state" (in the [File Options tab of Preferences](#)).

#### Tailing on Network Drives

Tailing relies on notifications from the operating system whenever a file has been modified. These notifications are not always sent for network drives. In particular, they are usually not sent for Unix network drives. Under such circumstances, file tailing will not work.

#### **Note**

File Tailing only applies if data is added to the end of the file. If any other part of the file is updated, **V** will not automatically update it.

#### 4.3.174.2.20 Viewing the Clipboard Contents

The File menu contains a View Clipboard submenu that displays the different types of data that are stored in the Windows clipboard (if any). Selecting the various data types will cause **V** to view the corresponding data just like it was viewing a file.

##### **Notes**

**V** can only view the clipboard data as text/hex. It cannot launch another program to view the clipboard data. For example, it cannot launch your image viewer to view bitmap data. However, you can [save the data to a file](#) and then use your image viewer to view that file.

Some data types cannot be viewed. If this is the case, the data type will be disabled on the menu.

The clipboard contents can be printed - just like a normal file.

#### 4.3.174.2.21 Customizing Colors

Select *Customize Colors* from the *View* menu to customize the colors that **V** uses to display the file.

Once **Use Default Colors** has been disabled, the following colors can be customized:

##### **Normal Background/Text**

The colors used to display the file.

##### **Highlighted Background/Text**

These colors are used to display highlighted/selected text. By default, these colors are the inverse of the text colors. That is, the background color is the text color and the text color is the background color.

**Search Line Background/Text**

When searching, the found text is displayed using the above **highlighted** colors.

Only the portion of the line that contains the found text will normally be highlighted. However, the part of the line that does not contain the found text may also be displayed in a different color to the standard text color. Doing this makes it easier to distinguish the line which contains the found text, especially if the found text is scrolled off the screen.

**Highlight All Background/Text**

These colors are used to display all string matches when the [Highlight All](#) option is enabled in the [Search Bar](#).

**Highlight All Line Background/Text**

These colors are used to display the non-highlighted portion of all lines that contain a [Highlight All](#) match.

**Line Numbers Normal/Highlight All**

The first color is used to display line numbers (and addresses in HEX mode). It is also used to display text in [fixed columns](#).

The second color will be used to display the line number for all lines that contain a [Highlight All](#) match.

**Current Line Background**

This color is used to draw the background of the *current* line.

**Bookmark Background**

This background color is used to display bookmarked lines.

**Grid Line**

This color is used to draw the Grid Lines when right-clicking on the [ruler](#).

**Block Marker**

This color is used to draw the start of block marker. A block marker is created by right-clicking on a position in the file and selecting [Mark Block->Start Point](#).

#### 4.3.174.3 Split File View

The File window can be split in two by clicking on the Split Screen button on the toolbar and selecting Horizontal or Vertical Split Mode. It can also be split from the Split submenu of the View menu, or by pressing Alt+S. This allows you to view different parts of the same file in different windows. Below is an example of a file which has been split horizontally.

|    | Date       | Open  | High  | Low   | Close | Volume    | Adj Close |
|----|------------|-------|-------|-------|-------|-----------|-----------|
| 1  | 2008-12-15 | 14.75 | 14.94 | 14.32 | 14.55 | 68647200  | 14.59     |
| 2  | 2008-12-12 | 15.78 | 14.85 | 13.69 | 14.75 | 68873000  | 14.75     |
| 3  | 2008-12-11 | 14.24 | 14.68 | 13.98 | 14.81 | 73845300  | 14.81     |
| 4  | 2008-12-10 | 14.26 | 14.88 | 13.75 | 14.27 | 74880000  | 14.27     |
| 5  | 2008-12-09 | 13.54 | 14.69 | 13.48 | 14.38 | 88120000  | 14.38     |
| 6  | 2008-12-08 | 13.56 | 14.81 | 13.45 | 13.94 | 78842400  | 13.94     |
| 7  | 2008-12-05 | 12.71 | 13.35 | 12.38 | 13.29 | 85179000  | 13.29     |
| 8  | 2008-02-15 | 20.31 | 20.54 | 19.98 | 20.11 | 78256600  | 19.70     |
| 9  | 2008-02-14 | 20.96 | 20.99 | 20.46 | 20.45 | 71565700  | 20.84     |
| 10 | 2008-02-13 | 21.06 | 21.28 | 20.86 | 21.21 | 49788900  | 20.77     |
| 11 | 2008-02-12 | 20.78 | 20.99 | 20.56 | 20.98 | 57734900  | 20.47     |
| 12 | 2008-02-11 | 20.48 | 20.82 | 20.15 | 20.68 | 53777900  | 20.25     |
| 13 | 2008-02-08 | 19.96 | 20.33 | 19.95 | 20.27 | 54170000  | 19.85     |
| 14 | 2008-02-07 | 19.66 | 20.36 | 19.50 | 20.05 | 76942200  | 19.64     |
| 15 | 2008-02-06 | 20.14 | 20.43 | 19.98 | 19.92 | 73305500  | 19.51     |
| 16 | 2008-02-05 | 20.69 | 20.87 | 20.18 | 20.12 | 85581600  | 19.71     |
| 17 | 2008-02-04 | 21.74 | 21.75 | 21.16 | 21.20 | 65724200  | 20.64     |
| 18 | 2008-02-01 | 21.40 | 21.82 | 21.22 | 21.77 | 66887600  | 21.19     |
| 19 | 2008-01-31 | 20.44 | 21.34 | 20.36 | 21.18 | 180753400 | 20.54     |
| 20 | 2008-01-30 | 20.36 | 21.12 | 20.29 | 20.69 | 96055500  | 20.14     |
| 21 | 2008-01-29 | 20.44 | 20.61 | 20.25 | 20.58 | 78365800  | 19.96     |
| 22 | 2008-01-28 | 19.88 | 20.38 | 19.62 | 20.23 | 68833200  | 19.75     |
| 23 | 2008-01-25 | 21.27 | 21.28 | 20.98 | 20.08 | 184701200 | 19.47     |
| 24 | 2008-01-24 | 20.15 | 20.74 | 20.07 | 20.68 | 98385900  | 20.14     |
| 25 | 2008-01-23 | 18.33 | 20.21 | 18.24 | 19.98 | 173781800 | 19.45     |
| 26 | 2008-01-22 | 18.28 | 19.08 | 18.05 | 18.61 | 146951600 | 18.12     |
| 27 | 2008-01-18 | 19.39 | 19.65 | 18.95 | 19.08 | 143864100 | 18.50     |
| 28 | 2008-01-17 | 20.02 | 20.85 | 19.21 | 19.53 | 172761000 | 18.82     |
| 29 | 2008-01-16 | 20.03 | 20.39 | 19.78 | 19.88 | 309347600 | 19.35     |
| 30 | 2008-01-15 | 23.00 | 23.28 | 22.51 | 22.65 | 129312400 | 22.09     |
| 31 |            |       |       |       |       |           |           |
|    | 0          | 10    | 20    | 30    | 40    | 50        | 60        |

Lines 213 to 235    96%    File Size: 12.5 KB (244 lines) [17/12/2008 13:39]    ANSI

Synchronized scrolling is enabled by default. [Click here for further details on synchronized scrolling.](#)

Note that both windows must use the same display mode. For example, you cannot have one window in hex mode and the other in text mode. Also, if you enable line wrapping, the wrapping will apply to both windows.

#### 4.3.174.3.1 Synchronized Scrolling

By default, synchronized scrolling is enabled when the file view is split.

For example, if the file is split horizontally, you can use either of the horizontal scrollbars to scroll both views simultaneously (the vertical scrollbars work independently of each other). Similarly, if the file is split vertically, either of the vertical scrollbars can be used to scroll both views vertically.

You can disable synchronized scrolling by pressing the *Split Screen* button on the toolbar and selecting *Split Options*. This allows you to modify the following options:

##### **Disable Horizontal Synchronized Scrolling (in Horizontal Split Mode)**

This disables horizontal synchronized scrolling when the file view is split horizontally. When enabled, the horizontal scrollbars will work independently of each other. Note that vertical scrolling is never synchronized when the file view is split horizontally.

##### **Disable Vertical Synchronized Scrolling (in Vertical Split Mode)**

This disables vertical synchronized scrolling when the file view is split vertically. When enabled, the vertical scrollbars will work independently of each other. Note that horizontal scrolling is never synchronized when the file view is split vertically.

#### 4.3.174.4 Multiple File Windows

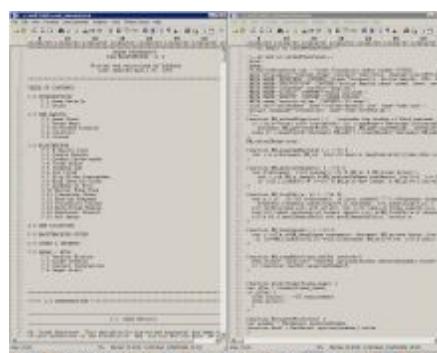
By default, every file you view will be displayed in a separate window. If you prefer to only have one file window open at a time, enable "[Use existing File window to view new file](#)" in the Window Layout tab of [Preferences](#).

##### 4.3.174.4.1 Arranging File Windows

By default, when you view multiple files, the file windows will be positioned so that the new file window slightly overlaps the previous window. If you rearrange the multiple window position, **V** tries to maintain this position when it opens multiple files.

You can tile the files by selecting the desired tiling option from the Window Layout menu (Vertical, Horizontal or Auto-Arrange). If you tile the files vertically, each file window will have the same height as the screen. Each window will have the same width and be placed next to each other so they completely fill up the screen.

The image below shows 2 file windows, tiled vertically.



### Automatic Tiling

Multiple file windows can be automatically tiled by enabling "[Automatically Tile multiple file windows](#)" in the [Window Layout](#) tab of [Preferences](#). That is, once a second (and subsequent) file is opened, **V** will automatically tile all the file windows.

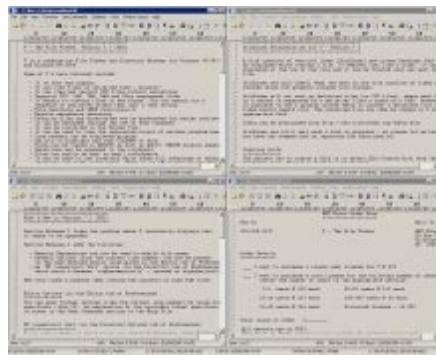
File windows can also be "Auto-Arranged". [Click here for further details](#).

#### 4.3.174.4.2 Auto-Arranging File Windows

File windows can be auto-arranged based on a "grid size" which you define in the [Window Layout](#) tab of [Preferences](#).

To see how auto-arranging works, consider a grid size of 3 x 3. With 2 or 3 windows open, they will be tiled vertically. When the user opens up a fourth file, they will be displayed in a 2 x 2 grid (as in the image below). When a user opens up a fifth and sixth file, the first 3 files will be displayed on the top and the remaining 2 (or 3) files will be displayed on the bottom. If the user opens up a seventh file, **V** will display the files in 3 rows of windows. The first 2 rows will contain 3 windows each, and the third will contain the seventh file. The eighth and ninth file will also be displayed on the 3rd row.

Because the grid size is only 3 x 3, the automatic arranging will stop with the ninth file. Should the user open up a tenth file, it will be displayed in a normal overlapping window.



#### 4.3.174.4.3 Synchronized Scrolling

When multiple files are open, scrolling can be *synchronized* by right-clicking on the scrollbar and selecting *Synchronize Scrolling*. Once enabled, whenever one window is scrolled (either via the keyboard or the mouse), all other file windows are also scrolled by the same amount.

Horizontal and vertical scrolling are *separate*. Enabling synchronized scrolling by right-clicking on the vertical scrollbar will only synchronize vertical scrolling. If you also want horizontal scrolling to be synchronized, you also need to right-click on the horizontal scrollbar and select *Synchronize Scrolling*.

#### Notes

Synchronized scrolling only applies to multiple file windows - it does not apply to the directory listing.

This is different to the synchronized scrolling which is available when the [file view is split](#).

#### 4.3.174.5 Hex Mode

Files displayed in hex mode look something like this:

|          | 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F | 0123456789ABCDEF |
|----------|-------------------------------------------------|------------------|
| 00000000 | 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 | MZ.....          |
| 00000010 | B8 00 00 00 00 00 40 00 00 00 00 00 00 00 00 00 | .....@.....      |
| 00000020 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | .....            |
| 00000030 | 00 00 00 00 00 00 00 00 00 00 80 00 00 00 00 00 | .....,..         |
| 00000040 | 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 | ..  .. =!@.L=!Th |
| 00000050 | 69 73 20 70 72 6F 67 72 61 60 20 63 61 6E 6F    | is program canno |
| 00000060 | 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 | t be run in DOS  |
| 00000070 | 60 6F 64 65 2E 00 00 0A 24 00 00 00 00 00 00 00 | Mode....\$.....  |

Hex mode displays a file as a series of hexadecimal (base 16) numbers together with the corresponding ASCII character equivalent (this is also known as **Debug Format**).

The first 8 digits on each line represents a hex address which indicates the position (or offset) of the corresponding line in the file. This is followed by up to 16 hex numbers (or bytes) which correspond to the file data.

The right hand side of the view consists of the ASCII character representation of the corresponding file data. If the hex byte does not correspond to a **printable** ASCII character, it is displayed as a "." (dot).

If the [Display ALL hex codes](#) option is enabled, all codes will be displayed on the right hand side instead of a dot. The character displayed depends on the selected font, and will usually not be unique for each control character.

Click here for a description of the various [hex formats](#) available.

### Note

By default, each line displays 16 bytes of data, although [the line length can be changed](#)

#### 4.3.174.5.1 Hex Formats

When in Hex Mode, the file may be viewed in one of several hex formats which are selected from the View menu (under Hex Formats) or from the right-click menu (Layout->Hex Formats).

The available formats are:

##### **Byte**

This is the default format. Each character in the file is displayed as an individual hex code (from 00 to FF).

##### **Decimal**

As above, but each byte is displayed in DECIMAL (from 0 to 255).

##### **Octal**

As above, but each byte is displayed in OCTAL (from 0 to 377).

##### **Word**

The data in the file is displayed as 16-bit words (always in hex)

##### **DWord**

The data is displayed as 32-bit double words (in hex)

##### **Double DWord**

The data is displayed as 64-bit quad-words (in hex)

##### **Flip Ends**

This only applies when in Word, DWord or Double DWord mode. The "ends" of each "word" are "flipped". This makes it easier to view data that is stored in little-endian format.

By default, each line displays 16 bytes of data, although the [line length can be changed](#).

##### **Note**

The file offsets are always displayed in Hex.

#### 4.3.174.5.2 Hex Line Length

By default, each line in hex mode contains 16 bytes.

This can be changed by selecting "Hex Formats"->"Set Hex Line Length" from the View menu (or pressing Ctrl-W).

The line length must be a multiple of the format size.

For example, if the data is being displayed as "words", the length must be a multiple of 2 (the size of a word).

#### 4.3.174.5.3 Hex Font

A different font can be used to display files in text and hex modes, although by default, these fonts are the same.

If you change the font in hex mode, the font will not be changed in text mode.

#### 4.3.174.5.4 Vertical Hex Mode

Vertical Hex Mode is enabled by pressing Alt+J (or selecting **Text + Vertical Hex** from the View menu). Vertical Hex mode looks as follows:

| 0                                        | 10        | 20        | 30        |
|------------------------------------------|-----------|-----------|-----------|
| 123456789                                | 123456789 | 123456789 | 123456789 |
| 1: 0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ |           |           |           |
| 3333333333244444444444444555555555555    |           |           |           |
| 01234567890123456789ABCDEF0123456789A    |           |           |           |

The file is displayed one line at a time (just as in text mode). However, each line is followed by 2 lines containing the hex code of each character in the line.

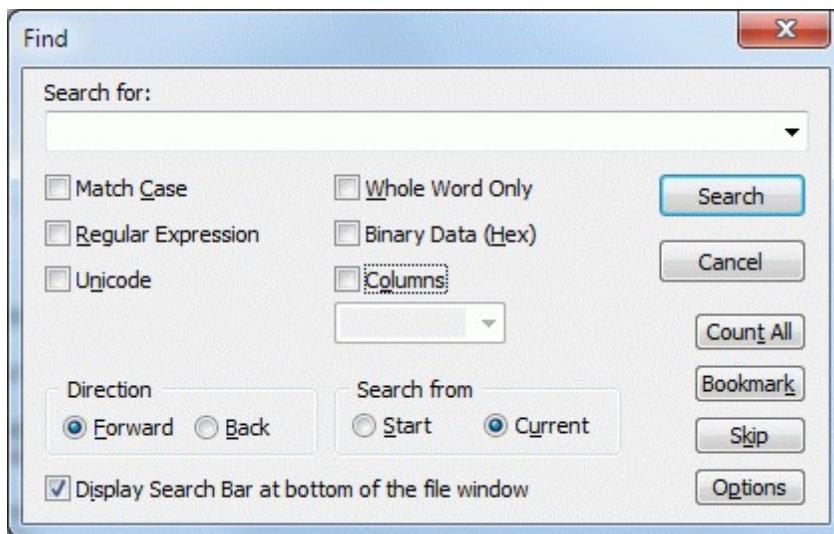
The 2 lines need to be read **vertically**. That is, the first line contains the first nibble (4 bits) of the code and the second line contains the second nibble.

In the example above, the first character in line 1 is ZERO, which is **30** in hex. Therefore, the first character in line 2 is **3** and the first character in line 3 is **0**.

#### 4.3.174.6 Searching

The **Find** command allows you to search for a string (or sequence of bytes) in the file being viewed. It may be selected from the toolbar, the Edit menu, from the right-click menu or from a [keyboard shortcut](#).

When you select the Find command, you will be presented with the Find dialog box, where you can specify your search. The Find dialog box contains the following:



Note that a [Search Bar](#) can also be displayed at the bottom of the file window. [Click here for details.](#)

### **Search for**

Enter the string to search for.

[Click here for several different ways of entering the search string.](#)

### **Match Case**

Usually **V** will ignore case (upper and lower) when searching for strings. Check this option if you only want **V** to find strings that exactly match the case of the string entered. Note that this option is disabled if the **Binary Data** option is checked.

### **Whole Word Only**

Check this option if you are only interested in matching the entered text when it appears as a word. For example, searching for the string "**soft**" will usually find a match in "**software**". Setting this option, will not find a match in "software".

### **Regular Expression**

This option indicates that the string specified in "Search For" is a regular expression.

### **Binary Data (Hex)**

This option indicates that the search string is a sequence of hex bytes rather than a text string.

**Direction**

This specifies the direction of the search. If **Forward** is selected, the file is searched from the beginning to the end. If **Back** is selected, the search begins from the end of the file and goes backwards.

**Unicode**

This will search for the Unicode equivalent of the specified string. This is useful if you want to search for strings in a Win32 executable. If the file being viewed is a [Unicode file](#), there is no need to enable this option.

**Search from**

Specifies from where the search is to commence. If **Start** is selected, the search will begin from the start of the file. If **Current** is selected, the search will begin from the current line. If the direction of the search is backwards, the Start option will be replaced with End. By selecting **End**, the search will commence from the end of the file.

**Columns**

You may restrict the search to a particular column or range of columns. [Click here](#) for further details on specifying column ranges.

**Flip Search String**

When viewing hex files in [flipped mode](#), this option is available to also flip the entered search string before searching.

**Display Search Bar at bottom of the file window**

This will display a [Search Bar](#) at the bottom of the file window that can be used instead of the Search dialog for searching.

Press the **Count All** button to [count the number of times](#) the search string appears in the file.

Press the **Bookmark** button to [bookmark the lines](#) that contain the search string.

Press the **Options** button to [modify the search options](#).

The **Search Again** command (or **Find Next**) will search for the next occurrence of the entered text. [Use Alt+A to continue the Find Next onto the next file](#).

### Cancelling the Search

If **V** has not found the specified string within 5 seconds, it will display a dialog box which will let you cancel the search by pressing the "Cancel" button.

### **Notes**

The Find Next command will always search forwards even if the previous search was backwards.

Ctrl+F3 can be used to search for the next occurrence of any highlighted text.

The search history is limited to **5** strings if **V** is not registered.

#### 4.3.174.6.1 Search String

In its simplest form, a search string consists of a sequence of text characters.

By enabling the **Binary Data (Hex)** option, the search string is treated as a sequence of hex bytes (eg, **FF096C3A**). Do not enter any spaces between the hex bytes.

You can also use specify hex characters within a text string by using the **\x hex notation**. For example, the above sequence of hex bytes can also be entered as:

"\xff\x09\x6c\x3a"

If you use the \x notation, you will need to first enable "[Allow hex characters in text search string](#)" in the [Search Options](#).

#### Notes

Case is not significant when entering hex data

When using \x notation, do not enable the Binary Data option

If \x notation is enabled and you actually want to search for "\x", you will need to enter "\\x"

#### 4.3.174.6.2 Search Options

Pressing the **Options** button in the Find dialog box will allow you to modify several search options.

##### **Find Next from Current Position**

The **Find Next** (or Previous) command will find the string immediately after (or before) the last string found - even if you have changed the file position. Enable this option to commence the search from the current file position (instead of the previous match).

##### **Find Next from next line**

By enabling this, **V** will ignore the rest of the line when doing a "Find Next" and will begin searching from the start of the next line.

##### **Wrap to Start**

By enabling this, **V** will continue searching from the start of the file once the end of the file is reached.

##### **Do not center found text**

This causes found text to always be displayed on the top line of the window - instead of being centered.

##### **Do not center found text if it is already on screen**

By enabling this option, if the string you are searching for is already visible, the window will not be scrolled when the string is highlighted.

##### **Only "beep" if search fails**

Enabling this will stop **V** displaying a "String not Found" message when no match is found. Instead, a short beep will sound.

##### **Always Start Ctrl-F search from the beginning**

By default, pressing Ctrl-F (or starting a search) will start the search from the [current position](#). Enabling this option will cause the search to always start from the beginning of the file.

##### **Allow hex characters in text search string**

This enable the use of the \x prefix to indicate a hex character when entering text strings.

##### **Disable "Match Case Toggling" (using the \ key)**

This disables [Match Case Toggling](#). That is, the \ key behaves just like the / key.

#### 4.3.174.6.3 Search Count

Pressing the **Count** button in the Find dialog box will count the number of times the search string appears in the file.

The number of matches found will be displayed on the screen as **V** is counting. Press the Cancel button to stop the count.

If any text is selected before starting the count, the number of times the search string appears in the selection will also be displayed

**Note:** Each instance of the specified string is not highlighted after the count is complete.

#### 4.3.174.6.4 Search Skip

Pressing the **Skip** button in the Find dialog box lets you skip over a specified number of matches.

For example, you can use this to find the 100th occurrence of a string in a file or to skip over 100 matches before continuing your search.

When the button is pressed, simply enter the number of matches to skip and press OK.

#### 4.3.174.6.5 Column Search

When searching a file for text, you may restrict your search to a column or range of columns.

To do this, you must enable the **Columns** checkbox and enter the column (or range) in the space provided. The most recently used columns may be selected from the drop-down list box.

Only text that begins in the column (or lies in the column range) will be matched.

You may specify a column range in one of 2 ways:

- n-m** Match strings that start anywhere between columns n and m
- n-** Match strings that start anywhere after (and including) column n

You may also specify more than one column range by separating them with commas. The following are examples of valid column specifiers:

|                      |                                                                                                                 |
|----------------------|-----------------------------------------------------------------------------------------------------------------|
| <b>1</b>             | Match if text starts at column 1 (ie, start of the line)                                                        |
| <b>1-10</b>          | Match if text starts in columns 1 to 10                                                                         |
| <b>1,12,80</b>       | Match if text starts in column 1 or 12 or 80                                                                    |
| <b>1-5,20-29,80-</b> | Match if text starts in columns 1 to 5, or columns 20 to 29, or starts anywhere after (and including) column 80 |

#### [Using Regular Expressions to search for data in columns](#)

Regular expressions can also be used to search for text in columns (and even for text not in a certain column).

#### 4.3.174.6.6 Tips on using the Keyboard

The following keys can be used to initiate a search:

##### **Normal Find**

/ and Ctrl+F.

##### **Find Next**

F3, A, Ctrl+L, Ctrl+N

You can also use the SPACE key if [Use SPACE as Find Next](#) is enabled in the [Keyboard tab of Preferences](#).

Ctrl+F3 can be used to search for the next occurrence of any highlighted text.

Alt+A (and Alt+F3) will [continue searching the "next file"](#) if no further matches are found in the file being viewed.

##### **Find Previous**

To search backwards, use any of the Find Next keys with the SHIFT key pressed.

##### **Backwards Search**

Using ? will always initiate a backwards search. That is, a normal search with the Direction option set to Back.

##### **Match Case toggle**

Using \ will initiate a search with the default "Match Case" option toggled.

That is, if you perform a normal search (eg, using /) with the Match Case option disabled, pressing \" will initiate a search with the Match Case option disabled. If you perform a normal search with the Match Case option enabled, using "\"" will disable the Match Case option.

You can disable this behavior by enabling [Disable "Match Case Toggling" \(using the \ key\)](#).

#### 4.3.174.6.7 Scrolling

The File View may be scrolled in the standard ways - via the keyboard or the scroll bars.

##### Using the keys

###### **PageUp and PageDown**

Scroll the view one screen-full at a time in the specified direction.

###### **UpArrow and DownArrow**

Scroll the view vertically one line at a time.

###### **LeftArrow and RightArrow**

Scroll the view horizontally, one column at a time.

###### **Ctrl-LeftArrow and Ctrl-RightArrow**

Scroll the view horizontally by the width of the view.

###### **Ctrl-Home and Control-End**

**Ctrl-End** scrolls the window so the end of the current line is visible. **Ctrl-Home** scrolls the window to its leftmost position.

###### **Home/End**

The **Home** key goes to the start of the file and the **End** key goes to the end of the file.

If the [Use Space as PgDown option](#) is enabled, the **SPACE** key will behave like **PageDown** and **Shift-SPACE** will behave like **PageUp**.

##### Using the scroll bars

If you click on the arrows on either side of the scroll bars (vertical or horizontal), the view is scrolled by one line (or column) in the appropriate direction. Clicking on the area below the vertical scroll box (or slider) is equivalent to pressing **PageDown** and clicking on the area above the scroll box is the same as pressing **PageUp**. Similarly, clicking to the right of the horizontal scroll box will scroll the view to the right by the width of the view and clicking to the left will scroll the view to the left.

Dragging the vertical slider allows you to quickly move the file position. You may enable [smooth scrolling](#) in order to improve the file display while dragging.

##### Current Line Marker

Pressing the Up/Down arrow keys will move the current line marker (instead of scrolling the window). The window is only scrolled when the line marker reaches the bottom (or top) of the window. If you would like the window to always scroll on an arrow key, enable [Always scroll window when using arrow keys](#) in the [File Options tab of Preferences](#).

If the window is scrolled using the scroll bars (with the mouse), **V** will automatically move the current line marker accordingly. If you prefer the current line marker to stay where it is, enable [Do not move current line marker when using scroll bars](#).

### Continuous scrolling

The view may also be scrolled continuously in the vertical direction. This is basically equivalent to continually pressing the UpArrow or DownArrow key.

To commence continuous scrolling, simply press **Ctrl+Shift+DownArrow** or **Ctrl+Shift+UpArrow** depending on the direction in which you want to scroll. The view will then start to scroll automatically. The speed of the scrolling may be increased by pressing the **+** (**PLUS**) key and decreased by pressing the **-** (**MINUS**) key.

Continuous scrolling may be stopped by pressing the **ESC** key and paused by pressing **SPACE**. Once paused, the scrolling may be re-started by pressing **SPACE** again, or terminated by pressing **ESC**. The scrolling will stop when the top (or bottom) of the file is reached.

### IntelliMouse Support

V makes full use of the Microsoft IntelliMouse. [Click here for details.](#)

### Customizing the keys

All of the cursor keys can be customized by selecting [Customize Keyboard from the Tools menu](#). For example, you can change the behavior of the Home/End keys so they go to the start/end of the line instead of the start/end of the file. The commands corresponding to the cursor keys (such as Line Up/Line Down) can be found in the *Other* submenu of the FILE commands.

#### 4.3.174.6.8 Smooth Scrolling

The slider on the vertical scrollbar may be dragged to quickly change the position of the file. **V** will continually try to redisplay the file as the scrollbar is being dragged.

If you find the screen refresh during scrolling annoying, you may try to make it "smoother" by enabling "Smooth Scrolling" in the [File Options section of Preferences](#).

To do this, you must specify a **Smooth Delay** which will be used to "slow down" the speed of the scrolling. Typically, a delay of between 50 and 200 will be used. If no delay is entered, **V** will use a delay of 180.

The delay specified will depend on the speed of your system. If you find the scrolling too slow, enter a smaller delay. If the screen "flashes" too quickly during scrolling, enter a larger delay.

#### Note

The **V** smooth scrolling is different from, and independent of the "smooth scrolling" which is available in programs like Internet Explorer.

#### 4.3.174.6.9 IntelliMouse Support

The Microsoft IntelliMouse is a mouse with a small wheel between the two mouse buttons. The wheel can be used to scroll windows without having to move the cursor over the scroll bars.

**V** supports the IntelliMouse as follows:

When viewing a file, scrolling the IntelliMouse wheel will scroll the document three lines at a time.

- If the SHIFT key is pressed while scrolling the wheel (or if the wheel itself is pressed), the document will be scrolled a screen at a time.
- If the wheel is scrolled while it is pressed and the SHIFT key is also pressed, continuous scrolling will begin. Pressing the wheel will pause/restart the scrolling. Pressing the wheel while the SHIFT key is pressed will stop the scrolling.
- If the CONTROL key is pressed as the wheel is scrolled, the Previous/Next document in the File List is displayed.

#### Note

In order for the operations requiring a wheel press to work correctly, you must "Turn on the wheel button" in the IntelliMouse setup (in the Control Panel) and set the "Button Assignment" to "Default".

#### 4.3.174.6.10 Goto

The Goto dialog box allows you to specify a location in the file to jump to. The specified location will be displayed at the very top of the File View. You may enter the location either as a **Line Number**, **Column Number**, **Offset**, **Page#**, **Record#** or **Chunk** by selecting the appropriate option.

Note that **Page#** is only enabled if the file is [paginated](#).

### Record Numbers

A **record** is different from a **line** in that it is always a fixed length (and it doesn't have to end with a newline character).

If the file being viewed consists of fixed length records, the record length will automatically be placed in the **Length** field. If the file does not consist of fixed length records, you may specify your own record length (although it probably wouldn't make much sense to do this).

The other options in the Goto dialog box are:

#### Hex

This specifies that the Offset entered is in hex instead of decimal.

#### From End of File

This indicates that the specified location is to be treated as being from the end of the file. For example, if you enter a Line Number of 100, **V** will position the view 100 lines from the end of the file.

#### **Notes**

It is quite valid to goto a file offset while in text mode. In this case **V** will simply goto the line which corresponds to the offset entered.

On the other hand, it is not possible to goto a particular line number if the file is opened in Hex mode (unless the file has also been opened in Text mode). In this case, the Line Numbers option in the Goto dialog box is disabled.

If the file is being displayed in [chunks](#), the numbers in the Goto dialog box are relative to the **start** of the file. [Click here for further details](#).

#### 4.3.174.6.11 Goto and Chunks

All entries in the Goto dialog are relative to the **start** of the file, regardless of which chunk is currently being viewed.

For example, if you were viewing the last chunk of a file, going to line 1 will take you to the very first line of the file (not the first line of the chunk).

You can also enter numbers outside the current chunk.

For example, if you were viewing the first chunk (which contained 1000 lines), you could enter line 5000 in the Goto dialog.

In this case, **V** would load the chunk that contained line 5000.

Note that **V** may take some time to locate the required position (especially if you are viewing a very large file). If the time taken is more than five seconds, a Cancel button will be displayed which will allow you to stop the goto operation.

#### **Note**

This behavior was introduced in Version 7. In prior versions, the line number was restricted to the current chunk.

#### 4.3.174.6.12 Bookmarks

Bookmarking allows you to remember the current file position so that you can easily return to it.

**V** implements two types of bookmarking - **numbered** and **traditional**.

Numbered bookmarking allows you to create up to 10 bookmarks (numbered from 0 to 9). [Click here for further details.](#)

Traditional bookmarking allows you to bookmark a line by pressing **Ctrl+F2**. Pressing Ctrl+F2 again will clear the bookmark. Unlike numbered bookmarks, there is no limit to the number of traditional bookmarks you can have.

Bookmarked lines will be displayed in a different color. When multiple lines have been bookmarked, pressing **F2** will take you to the *next* bookmark in the file and pressing **Shift+F2** will take you to the *previous* bookmark in the file. All bookmarks in the file can be cleared by pressing **Ctrl+Shift+F2** or by selecting *Bookmarks->Clear All Bookmarks* from the *Edit* menu.

#### [Search and Bookmark](#)

The [Search dialog box](#) contains a *Bookmark* button. Pressing this button will bookmark all lines that contain the search string. If you are using the [Search Bar](#), you can right-click on the *Find Next* button (down arrow) to bookmark lines containing the search string. The number of bookmarked lines will be displayed in the status bar.

All bookmarked lines can be *copied to the clipboard* by selecting *Copy Bookmarked Lines* from the *Bookmarks* submenu of the *Edit* menu. All bookmarked lines can be saved to a file by selecting *Save Bookmarked Lines* from the same submenu.

By default, **V** will clear all bookmarks when it exits. If you want the bookmarks saved so they will be available every time you view the file, enable [Save Bookmarks on Exit](#) in the [File Options tab of Preferences](#).

#### Note

The color of the bookmarked lines can be changed by selecting *Customize Colors* from the *View* menu.

#### 4.3.174.6.13 Numbered Bookmarks

Up to 10 bookmarks (numbered from 0 to 9) may be set by either selecting the appropriate bookmark number from the **Edit->Bookmark->Set Numbered** menu, or by pressing **Alt-0** to **Alt-9** (for bookmarks 0 to 9 respectively). Bookmarked lines are displayed in a different color.

Once a numbered bookmark has been set, it may be restored by selecting it from the **Edit->Bookmark-> Goto Numbered** menu. Alternatively, it may be restored by pressing **Ctrl-0** to **Ctrl 9**.

Numbered bookmarks are *global*. That is, you can save a bookmark in one file and restore it while you are viewing another file.

#### Notes

Numbered bookmarks not only save the current file position, but they also save the current file mode (ie, text or hex).

Unlike traditional bookmarks, numbered bookmarks can not be saved when you exit **V**. That is, every time you start **V**, all numbered bookmarks will be cleared. If you want to save a numbered bookmark, you should use [Favorites](#) and assign a shortcut key to the Favorite created..

#### 4.3.174.6.14 Sending selected text to your browser

**V** does not underline hyperlinks in files - such as <http://www.fileviewer.com/>, or create clickable hotspots like a browser does.

However, if a text file contains a URL, you may open the URL in your browser (not in **V**) by highlighting the text that forms the URL, right-clicking and selecting **Open Selection in Browser**.

You may also simply right-click over the URL (without selecting the actual text).

#### Notes

You do not have to highlight the entire URL - any part of the URL (even just one character) is sufficient.

If you only want to send **part** of the URL to the browser, simply select exactly what you want sent, and then press the Shift key when selecting **Open Selection in Browser**.

#### 4.3.174.6.15 Open Selection in V

While viewing a file, you may highlight some text and then select "**Open Selection in V**" from the File menu. The selection will be treated as a file name, and if it exists, will be viewed by **V**.

Alternatively, you can simply right-click on any part of the file name (without having to highlight anything) and then select "Open Selection in V". Note that this will only work if the file name does not contain any spaces. If it does, you will need to highlight the entire file name.

#### 4.3.174.6.16 Maintaining File Position

When viewing a file, the file position when last viewed is restored. This means that if you are quickly swapping between 2 files (using the Previous/Next commands), the file positions will not be reset to the beginning of the file.

You can disable this behaviour by enabling the [Do not restore file position](#) option in the File Options tab of [Preferences](#).

Note that the file positions are only remembered while **V** is active - they are not saved when you exit **V**.

#### 4.3.174.6.17 Paginated Files

A **paginated** file is one that contains form feeds (ASCII 12).

**V** will automatically paginate files that contain form feeds, displaying a page marker (dotted line) before the first line in each page.

By enabling [Page Up/Down go to start of page](#) in the Keyboard tab of [Preferences](#), **V** will always scroll to the first line in a page when pressing Page Up/Down.

If you press Ctrl+Shift+Page Up/Down, **V** will scroll to the next page, but maintain the current position in the page. For example, if you are currently on line 10 of page 1, pressing Ctrl+Shift+Page Down will take you to line 10 of page 2.

Do not enable this option if you want Page Up/Down to behave normally (ie, scroll by the length of the window).

#### **Line Numbers**

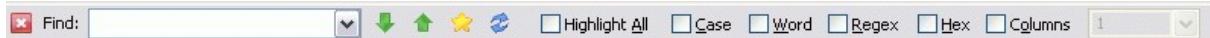
By default, **V** will increment the line number for each line in the file. If you want the line number of each page to always start at 1, enable the [Reset Line Numbers on New Page](#) option.

#### **Note**

If you don't want **V** to paginate the files, select [End Of Line->Ignore Form Feeds](#) from the View menu.

#### 4.3.174.6.18 The Search Bar

The Search Bar is displayed at the bottom of the file window by enabling the [Display Search Bar at bottom of the file window](#) option located at the bottom of the Search dialog box.



If the Search Bar is enabled, it will be used for searching instead of the Find dialog. To search, simply enter the search string in the "Find" box and press the Enter key. You can also click on one of the arrow buttons to search in the required direction (forwards or backwards).

If the search bar is not wide enough to display all of the search options, you can right-click on any empty area of the search bar to select the required option. This right-click menu also allows you to modify several other [search options](#).

After doing a search, you can click on the arrow buttons to do a [Find Next](#) (down arrow) or [Find Previous](#) (up arrow).

#### [String Count](#)

Right-click on the *Find Next Match* button (the down arrow) and select *Count All* to count the number of times the search string occurs in the file.

#### [Bookmarking](#)

Right-click on the *Find Next Match* button and select one of the Bookmark options to bookmark all the lines that contain the search string.

#### [Favorite Searches](#)

By clicking on the "Favorites Searches" button and selecting "Add Search", the current search is added to the list of Favorite Searches. The user can then perform this search by simply clicking on the Favorite Searches button and selecting the search from the list.

Organize/Configure can be selected from the Favorite Searches menu to modify the search or customize the menu (similar to how [Favorites are organized](#)).

#### [Highlight All](#)

Enabling [Highlight All](#) will highlight every occurrence of the search bar text (this does not work in hex mode). Note that the search is not incremental (as in Firefox). That is, it does not search for the next match as you type each character. You must click on the Refresh button (or Find Next) for the highlight/search to be updated.

#### [Search Bar Options](#)

You can customize some of the Search Bar functionality by right clicking on the Search Bar and selecting "Search Bar Options" or by clicking on the two small arrows at the very right of the search bar.

#### [Only display after first search](#)

By default, if the Search Bar is enabled, it will be displayed as soon as a file is viewed. By enabling this option, the Search Bar will be hidden when the user first views the file, and will only be displayed once

the user starts searching (by clicking the Find button in the toolbar or by using a search key).

#### **Do not use when searching from toolbar**

If the Search Bar is enabled, clicking on "Find" in the toolbar will move the cursor to the Search Bar for the user to enter the search text (instead of displaying the dialog box). By enabling this option, the default Find dialog will be displayed instead. If a search key is used (like '/' or Ctrl+F), the cursor will still be moved to the Search Bar.

#### **Automatically Search from Favorites**

Selecting a search from Favorite Searches only updates the search bar with the search text/options. The user still needs to click on a search button to perform the search. By enabling this option, the search is performed as soon as the favorite is selected.

#### **Automatically apply when file is loaded**

If Highlight All is enabled, enabling this option will automatically start highlighting text as soon as the file is loaded (using the most recent search string). If this option is not enabled, highlighting will only commence after the user does a search (or clicks on the Refresh button).

### **Colors**

Five colors can be defined in the Search Bar Options. These colors can also be set by selecting [Change Colors](#) from the View menu.

#### **Text/Background**

This is the color that will be used to display the "highlight all" text.

#### **Line Text/Background**

This color will be used to display any non-matching text on lines that contain highlighted text. This makes it easy to see what lines contain matches.

#### **Line Numbers**

This color will be used to display the line number of all lines that contain highlighted text.

#### 4.3.174.6.19 Continuing Search onto Next File

By default, **V** only searches the file being viewed for the search string. In particular, doing a Find Next will fail to find anything if no further matches of the search string can be found.

However, doing a Find Next using Alt+A (or Alt+F3), will cause **V** to continue searching the "next file" for further matches. If it finds any, it will load the new file and display the next match.

##### What is the "next file"?

You can always tell what the "next file" will be by placing your mouse over the Next File icon in the toolbar (the icon with the small right-arrow). Alternatively, you can click on the File List icon (small down-arrow) to display an ordered list of the files.

If you viewed the current file by selecting it from the **V** directory listing, the next file will be the next file in the directory listing.

If you launched **V** from the command line with more than one file (eg, **V \*.txt**), the file list contains all the files specified on the command line.

##### **Note**

You cannot search backwards across files.

#### 4.3.174.7 Block Marking / Text Highlighting

Several very useful operations can be performed on marked blocks (or selected text).

See the following sections for further details:

[Status Bar](#)  
[Selecting Words](#)  
[Marking Columns](#)  
[Adding Columns](#)  
[Copying to the clipboard](#)  
[Copying to a file](#)

Blocks can be marked in several ways:

**Using the Keyboard**  
[Click here for further details.](#)

##### **Mark with mouse**

A block may be marked in the usual method of holding down the left mouse button and dragging it over the text to be selected, releasing the button when the required text has been highlighted.

##### **Specifying the start and end positions**

This method is useful when marking very large blocks, or in hex mode, when you want to mark a specific address range.

To mark the start of a block, simply right-click on the corresponding character (or hex bytes) and select **Mark Block->Start Point**. To select the end of the block, scroll to the appropriate position in the file, right click on the corresponding character and select **Mark Block->End Point**. Once the start and end points have been specified, **V** will highlight the appropriate area.

Note that the end of the block may be marked before the start. Also, once a block is marked, it may be modified by choosing a new start or end point.

**V** will draw a marker around the start (or end) of a character to indicate the start (or end) marker. This marker will remain visible until it is cleared (by right-clicking and selecting **Mark Block->Clear**) or moved (by selecting a different position).

Note that the color of the marker may be specified by selecting [Change Colors](#) from the View menu.

#### **Marking to End or Start**

This method is similar to the above, except you only need to specify one point in the file (by right clicking at the appropriate location). The second point will either be the end or the start of the file. To mark a block from a character position to the end of the file, right click on that character position and select **Mark Block->To End**. Similarly, to mark a block from the start of the file to a character position, right-click on the end character position and select **Mark Block->To Start**.

#### **Offset from current position**

You may mark an exact number of characters from a file position by right-clicking on the file position, choosing **Mark Block->Plus Offset** (or Minus Offset) and then specifying the number of characters to mark.

### **Marking the entire file**

The entire file may be marked by selecting **Select Block->Entire File** from the **Edit** menu, or by right-clicking and selecting **Mark Block->Entire File**. The entire file may also be selected by pressing **Ctrl-A**.

### **Selecting a word or line**

To highlight an entire word, simply double-click anywhere on the word. Usually, only alphanumeric characters and the underscore are used to determine what constitutes a word. To use a more extended character set to select a word, press the **shift** key while you double-click. For example, double-clicking at the start of "<http://www.fileviewer.com/>" will only highlight "http". However, by also pressing the **shift** key, the entire URL is highlighted. [Click here for further details on how to customize this behaviour.](#)

To select the entire line, simply press the **control** key while you double click anywhere on the line.

If you double-click on a word (to select it) and then move the mouse before releasing the left mouse button (to select further text), **V** will always end the selection at the end of a word.

### **Extending the selected text**

Once text has been selected, it can be extended (or shortened) by **shift-clicking** on the new end (or start) position. That is, click on the new end position while the **shift** key is pressed.

### **Marking Columns**

[Click here for details on how to select text in a particular column range.](#)

### **Notes**

- Pressing **Ctrl+F3** will search for the next occurrence of any highlighted text.
- When HEX data is highlighted, both the Hex and Character views of the file are highlighted.
- Clicking the left mouse button on any part of the file view will clear any marked block (but it will not clear any block marker).

#### 4.3.174.7.1 Using The Keyboard

Text selection via the keyboard is achieved as in most text editors - and usually involves using the cursor keys together with the Shift key. In particular:

**Shift+DownArrow** will select the current line.

**Shift+RightArrow** will increase the text selection by one character.

**Shift+End** will select to the end of the current line

**Shift+PageDown** will select an entire "page" of text

### Selected Text and Scrolling

Unlike most text editors, **V** does not clear any selected text when the keyboard is used for scrolling. That is, as you scroll the file with the arrow keys, any selected text remains selected. The selected text will remain selected until you either start a new text selection or you left-click on the file with the mouse.

The selected text can also be cleared by pressing the key associated with the *Clear Selected Text* command. There is no default keyboard shortcut to do this, but you can [customize the keyboard](#) to assign a key to *Clear Selected Text* (which is in the *Other* submenu of *FILE*).

#### 4.3.174.7.2 Status Bar

The bottom left section of the status bar will display details of the selected text as it is being highlighted.

#### In text mode it will display:

- The line and column number of the start and end positions of the selected text
- The total number of characters in the selected text (in decimal and hex)
- If the [Hex offset in status bar option](#) is enabled, the hex offsets corresponding to the start and end of the selected text will also be displayed.
- If [highlighting a column](#), **V** will also display the sum of the column data if the column consists of numbers.

#### In hex mode it will display:

- The start and end offsets of the selected text (in hex)
- The number of characters in the selected text (in decimal and hex)

#### 4.3.174.7.3 Selecting words

While viewing a file, **V** will highlight an entire word by double-clicking anywhere on the word. By default, a word is defined to consist of alphanumeric characters (ie, a-z, A-Z, 0-9) and an underscore . Pressing the shift key while double-clicking on a word uses an extended character set to determine what characters to highlight. By default, the extended character set consists of all alphanumeric characters and any character **not** contained in the following:

```
"'!$=;<>{}[](),
```

#### 4.3.174.7.4 Marking Columns

Usually when marking blocks that span several lines, complete lines in the block are highlighted - except for maybe the first and last line in the block, which may be partially highlighted.

Sometimes you may only want to highlight text that appears in certain columns. You may do this by pressing the **Control** key as you drag the mouse over the text to be selected. In this case, the highlighted area will form a rectangle consisting of only the required columns. Note that it is not necessary to continue pressing the **Control** key as you are marking the text - only when you begin to mark the text.

When you copy the column selection to the clipboard (or save it to a file), only the text in the highlighted columns is copied/saved.

#### Notes

- Column marking is not available in HEX mode.
- Once a column is marked, it may be extended by shift-clicking, as described in the [How to mark blocks](#) section.
- If the columns contain numbers, you can have V [add them together](#) and display the sum in the status bar.
- Although it is possible to mark columns if V is unregistered, you may only copy the selected text to the clipboard (or save it to a file) once V has been registered.

#### 4.3.174.7.5 Adding Columns

When [selecting columns](#) which contain numbers, **V** can add these numbers and display the sum in the status bar.

To do this, right-click on the bottom left of the status bar and select **Sum Column Data**.

If you want **V** to [automatically](#) display the sum as you are selecting the column, right-click on the status bar and enable **Auto-Sum**. You may copy the sum to the clipboard by right-clicking on the status bar and selecting **Copy Sum to Clipboard**.

By default, **V** looks for valid decimal numbers to add. If the numbers in the column are in hex, also press the SHIFT key.

#### Notes

- Any lines in the selection that do not contain valid numbers will be ignored.
- This only applies to selected columns. That is, text that has been selected with the Ctrl key pressed.
- When selecting very large columns of data, the auto-sum option may slow down the selection process.

#### 4.3.174.7.6 Copying text to the clipboard

The selected text may be copied to the clipboard by either pressing **Ctrl-C**, by selecting **Copy/Cut** from the **Edit** menu (the right-click menu) or by clicking on the Clipboard toolbar button.

If you are displaying the file with line numbers, they will not usually be copied to the clipboard. If you would like the line numbers included, you have to enable the [Include line numbers on copy to Clipboard](#) option in the [Preferences](#) Dialog box.

#### Copying in Hex mode

In hex mode, the data to be copied to the clipboard depends on how the data is selected. If the data is selected by drawing the mouse over the right side of the display, then only the data bytes will be copied (nulls will be ignored).

If the data is selected by moving the mouse over the left side of the display, the actual hex representation of the data (including the hex address) will be copied.

#### Notes

- The selected text will usually remain highlighted once it has been copied to the clipboard (or copied to a file). If you want the selection to be cleared once the selection has been copied, set the **Unmark block after copy** option in the [File Options](#) tab of the [Preferences](#) dialog box.
- When selecting the left side of a view In Hex mode, only complete lines are copied to the clipboard, regardless of how much of the line is highlighted. Even if only one byte in a line is highlighted, the entire line (and address) is copied to the clipboard. (This does not apply when selecting the right side of the view)
- The selected text may also be [appended to the current clipboard contents](#) by pressing the SHIFT key while doing the copy.

#### 4.3.174.7.7 Appending to the clipboard

The selected text may be **appended** to the clipboard by pressing the SHIFT key while doing the copy. That is, press SHIFT while clicking on the clipboard toolbar button, or by pressing Ctrl-Shift-C instead of Ctrl-C.

#### 4.3.174.7.8 Copying to a file

The selected text may be copied to a file by selecting **Select Block->Write to File** from the **Edit** menu or by selecting **Mark Block->Write to File** from the right-click menu.

Copying selected text to a file is similar to copying to the clipboard, however you are asked to specify a file name to copy the text to. If the file already exists, you have the option of appending to the file instead of over-writing it.

This feature makes it easy to extract several small portions from a large file and save them to a much smaller file (perhaps to be printed).

### File Encoding

When saving text files, select the type of encoding from the "File Encoding" drop-down list box. The File Encoding will default to Unicode. The encoding should be set to ANSI/ASCII to save the text in the more common single-byte ASCII format.

### Copying in Hex mode

Unlike copying to the clipboard, copying to a file in Hex mode does not copy the hex data as it is displayed, but as it is stored. That is, if you highlight 6 hex bytes and copy them to a file, the resulting file will contain exactly 6 bytes.

#### 4.3.174.8 GridLines

A **Grid** consists of vertical lines (**GridLines**) and column headings that can be displayed while viewing a file. It behaves just like a ruler. It can be displayed at the top of the file and it can be floated over any part of the file.

|    | 0<br>123456789 | 10<br>123456789 | 20<br>123456789                         | 30<br>123456789 | 40<br>123456789 | 50<br>123456789 | 60<br>123456789 | 70<br>1234567 |
|----|----------------|-----------------|-----------------------------------------|-----------------|-----------------|-----------------|-----------------|---------------|
|    | Surname        | First           | Address                                 |                 |                 |                 |                 |               |
| 1: | Madison        | Oscar           | Apartment 1102, 1049 Park Avenue        |                 |                 |                 | New York City   |               |
| 2: | Mulder         | Fox             | Apartment 42, 2630 Hegel Place          |                 |                 |                 | Alexandria      |               |
| 3: | Seinfeld       | Jerry           | Apartment 5A, 129 West 81st Street      |                 |                 |                 | New York City   |               |
| 4: | Ricardo        | Ricky           | Apartment 4A, 623 East 68th Street      |                 |                 |                 | New York City   |               |
| 5: | Richards       | Mary            | Apartment D, 119 North Weatherly Avenue |                 |                 |                 | Minneapolis     |               |
|    | Surname        | First           | Address                                 |                 |                 |                 |                 |               |

A Grid may be displayed by manually selecting it from the GridLines menu, or it can be automatically loaded whenever a particular file is viewed.

The state of the current Grid can be toggled by pressing Ctrl+Shift+G or by selecting Toggle Grid from the GridLines menu.

Note that GridLines are only useful when the data in the file consists of **fixed length records** which are properly **aligned** into columns.

GridLines will not work on delimited files (or CSV files), where each field in a record is separated by a delimiter (like a comma or a TAB). However, it is possible to use a program called **TuFix** to convert a delimited file into a fixed record length file. The converted file can then be viewed in **V** with an appropriate Grid.

Tufix can be downloaded from:

<http://www.fileviewer.com/TuFix.html>

Click on the following sections for further details on GridLines:

[Creating Grids](#)  
[Organizing Grids](#)  
[Wrap Options](#)  
[Automatically Loading Grids](#)  
[Associating Grids with a File Extension](#)  
[Exporting Grids](#)  
[Importing Grids](#)  
[Exporting Data to a CSV File](#)

#### **Note**

[GridLines can be printed](#) by using %g in the header/footer

#### 4.3.174.8.1 Creating Grids

To create a Grid, select **Edit/Create Grid** from the GridLines menu. This will display a blank Grid. After making the necessary modifications, you need to save the Grid before exiting.

To make modifications, **right-click** on the Grid and select the desired option. The following options are available:

##### **Add Grid Line Here**

Right-click at the position where you want to add a Grid Line and select this option. You can then use "Edit Column Name" to give the column a meaningful name. If you place the Grid Line at the incorrect position, you can simply left-click on the end of the column and drag it to the required position.

Note that you can also add a Grid Line by **Shift-Left-Clicking** at the required position.

##### **Edit Column Name**

Right-click on the appropriate column, then enter the column name (followed by the Enter key). You can leave the column name unchanged by pressing the ESCape key.

##### **Edit Column Length**

Right-click on the column and enter the new length. In most cases it would be easier to set the column length by simply left-clicking on the end of the column and dragging to the desired position.

##### **Select Font**

This lets you specify the Font to display the Column names. Note that all columns are displayed using the same font.

##### **Insert Column**

This always inserts a column of 10 characters just before the column that you right-click on. You can then drag the new column to the required position. In most cases, it would be easier to use "Add Grid Line Here".

##### **Delete Column**

Deletes the column that you right-click on.

##### **Hide Grid Lines**

This specifies that the vertical Grid Lines will not be drawn over the file. Only the Grid header will be displayed.

##### **Center Column Names**

The column names will be centered instead of starting at the left.

##### **Set Wrap Options**

This allows fixed length record files to automatically wrap to the correct record length when a Grid is loaded. [Click here for further details](#).

##### **Save**

This saves the Grid. If you are saving a newly created Grid (instead of an edited Grid), the [Organize Grid](#) dialog will be displayed, allowing you to enter a name for the Grid. This name will appear on the GridLines menu.

##### **Save As**

This lets you save the edited Grid with a new name. The Grid originally loaded will not be modified.

### **Export**

This lets you export the Grid to a .vgrid file. [Click here for further details.](#)

Note that exported Grids do not appear on the GridLines menu. If you want the Grid to also appear on the GridLines menu, you will have select Organize/Configure from the GridLines menu and then [import](#) the Grid.

### **Set Export Data**

This lets you specify additional data that will be exported to the .vgrid file.

[Click here for further details.](#)

### **Close Menu**

Select this if you do not want to make a selection from the right-click menu.

### **Exit Edit Mode**

Terminates the Grid edit. If you have not saved a modified Grid, you will be asked if you want to save it before exiting.

### **Notes**

You can also maintain Grids by selecting [Organize/Configure](#) from the GridLines menu.

You can only create/edit a grid while you are viewing a file.

Unregistered versions of V cannot save grids that have more than 3 columns.

#### 4.3.174.8.2 Organizing Grids

Select **Organize/Configure** from the GridLines menu if you want to re-organize your Grids.

Note that the easiest way of creating a Grid is to select **Create Grid** from the GridLines menu. The easiest way of modifying a Grid is to first load it, and then to select **Edit Grid** from the GridLines menu.

Using the Organize option to modify a Grid does not give you visual feedback as you are making the changes. You will need to exit the Organize dialog before any Grid modifications are displayed.

Organize/Configure is best used to make minor changes to the Grid, or to re-organize the Grids on the GridLines menu.

Organizing Grids is very similar to [Organizing Favorites](#). See the [Creating Grids](#) section for an explanation of the various options in the Organize dialog box. When organizing the grids, a shortcut key can be assigned so that the grid is selected whenever the corresponding keyboard shortcut is entered.

You can copy a Grid from another user by first [exporting](#) the Grid and then [importing](#) it.

#### 4.3.174.8.3 Wrap Options

When viewing files with fixed length records (and no line terminator), you need to manually enter the correct record length ([or wrap length](#)) for the file to be displayed correctly.

By entering the Wrap Options for a Grid, the file will be automatically wrapped to the specified record length when the Grid is loaded.

To enter the Wrap Options while creating/editing a Grid, right-click on the Grid header and select **Set Wrap Options**. The wrap options consist of the following:

##### **Wrap Lines at Column**

This is length at which all lines will be wrapped (ie, the record length).

##### **Restore wrap settings when Grid is removed**

By default, **V** will maintain the current wrap settings when the Grid is removed - even when you view a different file. If the wrap options do not apply to this file, you will need to disable (or change) them.

By enabling this option, **V** assumes that the wrap options only apply to the file being viewed and will restore the original wrap settings when the Grid is removed (or the file is closed).

#### 4.3.174.8.4 Automatically Loading Grids

A Grid is usually manually loaded by selecting it from the GridLines menu. If you always want a particular grid to be displayed when a file is loaded, you need to export the grid to a file name that is the same as the file with a **.vgrid** extension.

For example, to always display a grid with **Filename.dat**, you need to create a grid file with the name **Filename.dat.vgrid**.

The **.vgrid** file needs to be in the [same directory](#) as the file being viewed - or in the [Default Grid Directory](#) (see below).

##### **Creating a .vgrid File**

After you have [created a Grid](#), you can export it by selecting Export from the right-click Grid menu. Note that an exported Grid will not appear on the GridLines menu. If you want the Grid displayed in the GridLines menu, you will also need to Save it.

If you export the Grid to a file named **.vgrid** (just an extension, with no file name), that Grid will be used for **all files** in that directory.

##### **Default Grid Directory**

You can specify a [Default Grid Directory](#) by selecting [Default Grid Directory](#) from the GridLines menu. If a **.vgrid** file does not exist in the current directory, **V** will also look in this directory.

You can use [rules based loading](#) to load a grid based on [part of the file name](#). This makes it possible to load the same grid for multiple files without having to create multiple **.vgrid** files.

**Note:** In order to load grids automatically, you need to enable the **Auto-Load Grids** option on the

---

GridLines->Options menu.

#### 4.3.174.8.5 Using Rules (Regular Expressions) to Load Grids

If **Enable Grid Rules** is enabled (on the GridLines->Options menu), **V** will look for a **.rules** file which contains a list of file expressions and corresponding **.vgrid** files to load if the currently viewed file matches that expression.

You will need to create a file called **.rules** in the Default Grid Directory (select Set Default Grid Directory from the GridLines menu). This file must be created manually with an editor (you cannot use **V** to create it).

Each line in this file (**.rules**) consists of a regular expression followed by a replacement string (separated by a "/").

When viewing a file, **V** will try to match the file name against each expression in **.rules**. If it finds a match, it tries to load the **.vgrid** file specified by the replacement.

For example, let's say that a directory consisted of many files named **Test1-YYYY-MM-DD.log** and **Test2-YYYY-MM-DD.log** where YYYY-MM-DD represented the date the file was created.

You could apply a single **.vgrid** file for all such files by adding the following line to **".rules"**:

```
Test.*\log/Test.vgrid
```

This will cause **V** to load **Test.vgrid** for any file of the form **Testxxxxx.log**.

If you wanted to load a different grid for Test1 and Test2 files, you could add the following line to **.rules**:

```
Test([0-9]).*\log/Test\1.vgrid
```

This would load **Test1.vgrid** for all file names that started with **Test1**, **Test2.vgrid** for all file names that started with **Test2**, and so on, up to **Test9.vgrid**.

#### Notes

The expressions in **.rules** must be regular expressions - which are different from simple Windows/DOS wildcard expressions (like **Test\*.log** - which will not work).

Any lines in **.rules** that begin with either **#** or **;** are ignored (they can be used for comments).

#### 4.3.174.8.6 Associating with a File Extension

By associating a Grid with a File Extension, that Grid will be automatically loaded whenever a file with that extension is viewed.

To do this, simply create a Grid, and when saving it, make sure that the **name of the Grid** begins with the file extension.

For example, to associate a Grid with **.xyz files**, simply name the Grid something like "**.xyz (Grid for XYZ Files)**". Any description may follow the file extension (as long as there is a separating space).

You must also enable **Auto-Load Grid Extensions** on the GridLines menu.

#### Note

A File Extension Grid must be saved on the GridLines menu. You cannot save it as a .vgrid file in the current directory or the Default Grid Directory.

#### 4.3.174.8.7 Exporting Grids

Grids can either be exported to .vexp files or to .vgrid files.

#### Exporting to a .vexp file

Press the More button and select Export. This will allow you to export either the selected grid(s) or all the grids to a .vexp file. This is usually done so the grids can be copied by another user.

Once another user has imported the exported .vexp file, the imported gridlines will appear on their GridLines menu.

#### Exporting to a .vgrid file

Press the More button and select Export to .vgrid.

A Grid is exported to a .vgrid file so that it can be automatically loaded by V when a file is viewed.

The **Set Export Data** option in the Grid right-click menu is used to specify extra data that will be appended to **every line** in the exported .vgrid file.

You can also append different data for each line (column) by entering the extra data to be exported in the **Column Length**. Simply include the extra data **immediately after** the length.

For example, if you enter **16, A** for the Column Length, ", A" (without the quotes) will be added to the corresponding exported line.

Note that the comma in this example does not act as a separator. The extra data to be exported begins at the first non-numeric character after the column length. If you add a space after the column length, the space will also be exported.

#### Note

.vgrid files can only be exported/imported one at a time. Export to a .vexp file if you would like to export/import multiple grids.

#### 4.3.174.8.8 Importing Grids

When organizing GridLines, pressing the More button allows you to import GridLines that had previously been [exported](#). Two different types of import are available.

##### **Import from a .vexp file**

Selecting the Import option allows you import a previously exported .vexp file. The imported grids will appear on the GridLines menu.

##### **Import from .vgrid**

This allows you to add a Grid to the GridLines menu that has previously been [exported](#) to a .vgrid file.

##### **Note**

.vgrid files can only be exported/imported one at a time. Export to a .vexp file if you would like to export/import multiple grids.

#### 4.3.174.8.9 Exporting Data to CSV

The file being viewed can be exported to a CSV file by selecting *Export to CSV* from the *GridLines* menu. This option is only available if a Grid has been applied to the file being viewed.

Each line in a CSV file consists of multiple fields separated by a delimiter (typically a comma). Each field will contain the text in the corresponding grid column.

The following options can be set when exporting to a CSV file:

##### **Delimiter**

If nothing is specified in the Delimiter box, a comma will be used as the delimiter. If you want to specify another delimiter (eg, a vertical bar "|"), simply enter it here. Note that the delimiter can contain more than one character. If you want to use a TAB for the delimiter, enter "\x8" (without the quotes).

##### **Quotes**

By default, **V** will only place quotes around a field if the field contains a delimiter. You can change this behavior by specifying one of the following:

- Use Quote Always
- Use Quotes only if field contains delimiter
- Use Quotes only if field contains delimiter or spaces

##### **Do not strip trailing spaces**

By default, **V** will remove any trailing spaces from a field. Enable this option if you want the trailing spaces exported.

##### **Do not export grid headings**

**V** will export the grid headings to the first line of the CSV file. Enable this option if you do not want the headings exported.

##### **Note**

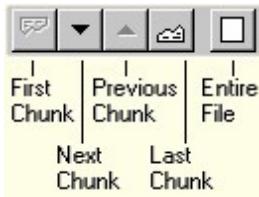
Only the first 500 lines are exported if **V** is unregistered.

#### 4.3.174.9 File Chunks

The time taken to load a file (in text mode) increases as the file gets larger. When viewing really large files, it is likely that you just want to look at the start and/or end of the file. In this case, it is pointless to try and load the entire file. Why load all of a 100Mb file, when you just want to view the last few lines?

In order to keep the load time to a minimum, **V** breaks the file into chunks. The size of the chunk is specified in the More Options tab of the [Preferences](#) Dialog box (it defaults to 8Mb). File Chunks are enabled by default - to disable them, just clear the Enable File Chunks option.

When a file is to be viewed, **V** checks the size of the file, and if it is larger than the chunk size, will only load the first chunk in the file (or the last chunk if you are viewing the tail).



An extra toolbar will appear in the top right of the window, which lets you load further chunks, and an extra status bar pane also appears in the bottom of the window, which displays which chunk is currently being viewed. You may also click on this area of the status bar to display available chunk options.

Once a chunk is loaded, the scrollbars restrict movement to within the chunk. That is, sliding the vertical scrollbar all the way to the bottom, will take you to the end of the chunk, not the end of the file. Also, the displayed line numbers are not always correct when viewing a file in chunks.

[Click here for further details on line numbering in chunks.](#)

Searching is not restricted to the current chunk. If a string is found outside the current chunk, then the appropriate chunk is automatically loaded.

In the [Goto Dialog Box](#), offsets and line numbers are relative to the start of the **file**, not to the start of the chunk. This means that if you were viewing the last chunk of a file, going to line 1 will take you to the very start of the file and not to the start of the chunk.

[Click here for further details on Goto and chunks.](#)

The buttons on the chunk toolbar have the following function:

**First Chunk** Loads the first chunk in the file. This will be disabled if the first chunk is the chunk currently being viewed.

**Next Chunk** Loads the next chunk in the file.

**Previous** Loads the previous chunk in the file.

**Last Chunk** Loads the last chunk in the file. This will be disabled if the last chunk is the chunk currently being viewed.

**Entire File** This will load the entire file, and the chunk toolbar will disappear. Note that this may take some time, depending on the size of the file. (See note below).

Pressing the **PageDown** and **LineDown** keys while at the very end of a chunk will automatically load the next chunk. Pressing the **PageUp** and **LineUp** keys while at the very start of a chunk will load the previous chunk.

### **Chunks in Hex mode**

File chunks do not usually apply in hex mode - the entire file is displayed, even if it is larger than the chunk size. However, if the file size is extremely large (usually > 2Gb), **V** will also use chunks in hex mode. This behaviour depends on available system memory and cannot be disabled.

### **Notes**

- The end of a chunk will usually fall somewhere in the middle of a line. **V** tries to break a chunk at the end of a line, however, this may not always be possible. Because of this, the last line of a chunk may sometimes be the same as the first line of the next chunk.
- You may go to an absolute chunk number by using the [Goto Dialog box](#).
- If **V** takes more than 5 seconds to load the entire file, it will display a progress dialog that will allow you to cancel the operation if you think it will take too long.

#### **4.3.174.1(EBCDIC Files**

When **V** opens a file, it tries to detect if the file is ASCII or EBCDIC. If it detects an EBCDIC file, it will automatically display it in EBCDIC mode.

If it incorrectly displays it as ASCII (text or hex), you can switch to EBCDIC mode by selecting EBCDIC from the View menu (or pressing Alt+B).

Similarly, if an ASCII file is incorrectly displayed as EBCDIC, you can return to ASCII mode by selecting ASCII from the View menu (or pressing Alt+B).

When a file is displayed in EBCDIC mode, **EBC** will be displayed on the bottom status bar. Clicking on **EBC** will allow you to modify several EBCDIC options.

[Click here for details on the various EBCDIC file formats supported](#)

#### 4.3.174.10.1 EBCDIC File Formats

If **V** does not display the EBCDIC file correctly, you can click on **EBC** on the status bar (or select EBCDIC Options from the View menu) to specify the correct file format.

EBCDIC files are usually in one of 4 formats.

##### **Carriage Return Delimited**

These files are just like ASCII files. That is, each line is terminated by a carriage return (or carriage return + line feed). The only difference is that the file contains EBCDIC characters instead of ASCII characters.

To display these types of files the **No Formatting (Display as EBCDIC file with CR/LF)** option should be enabled.

If some EBCDIC characters are not displayed correctly, you can modify the [EBCDIC to ASCII mapping](#) in the EBCDIC tab of [Preferences](#).

##### **Fixed Length Records (RECFM=F)**

Each line (or record) in the file consists of a fixed number of characters. To view these files, enable the RECFM=F option and enter the fixed record length in the **LRECL** field.

[Click here for further details on viewing RECFM=F files.](#)

##### **Variable Length Records (RECFM=V/VB)**

These files consist of variable length lines (or records). Each record is preceded by a 4 byte record descriptor which specifies the record length. **V** will automatically decode the records and display the lines as if they were delimited by a carriage return.

##### **Undefined Format (RECFM=U)**

By strict definition, the format of these files is **Undefined** (or **Unknown**). However in **V**, the meaning is slightly different. [Click here for further details.](#)

##### **Carriage Control (CCTYPE)**

[Click here for details on Carriage Control.](#)

##### **Use ASCII Character Set**

By default, EBCDIC RECFM files consist of EBCDIC characters. Enable this option if the file consists of ASCII characters.

##### **Trailing Spaces**

It is common for records/lines in EBCDIC files to be padded with trailing spaces - especially when files with variable length records are stored as RECFM=F.

Enable the **Ignore Trailing Spaces** option if you do not want the trailing spaces displayed.

#### 4.3.174.10.2 RECFM=F

**V** will try to "guess" if an EBCDIC file contains fixed length records (RECFM=F) and will try to guess the fixed record length.

However, it may sometimes get the record length wrong - requiring the user to select the EBCDIC Options and enter the correct record length.

If you do not want **V** to guess the record length, you can enable [Do not guess EBCDIC fixed record length \(RECFM=F\)](#) in the File Options tab of [Preferences](#).

If this is enabled, all fixed record length files will be displayed as [Carriage Return Delimited](#) files. To display as fixed record length files, you will need to:

1. Select EBCDIC Options from the View menu (or click on EBC in the status bar)
2. Disable the No Formatting option
3. Enable RECFM=F
4. Enter the correct record length in LRECL

#### Use Wrap To Length

Alternatively, you can make use of the [Wrap To Length](#) command to display the file without having to specify RECFM=F.

You can enable **Wrap To Length** in one of 3 ways:

1. By pressing the Wrap To Length icon on the toolbar
2. By selecting Wrap->Wrap To Length from the View menu
3. By pressing Alt+L

You will also need to specify the record length by selecting Wrap->Set Wrap Length from the View menu (or pressing Ctrl+W). **V** will remember the last wrap length used, so you will only need to set it if it has changed.

The disadvantage of using Wrap To Length to display RECFM=F files is that all files subsequently viewed will be wrapped to this length. That is, if they contain lines that are longer than the wrap length, they will be wrapped. You will need to remember to disable Wrap To Length after you have finished viewing the fixed record length file.

#### 4.3.174.10.3 RECFM=U

RECFM=U usually means that the file format is unknown or undefined.

However, I have seen files labeled as RECFM=**U** which are very similar to RECFM=**V** files - the only difference being that they have a 2 byte descriptor length instead of a 4 byte descriptor length.

**V** too will refer to these files as RECFM=**U**.

If you have a file that is in this format (that is, each record in the file consists of a 2 byte length followed by the record data), just set RECFM to **U** and the file should be displayed correctly.

**V** may not automatically recognize RECFM=U files. If this is the case, you will have to manually set the format to RECFM=U - which can be done in one of 2 ways:

##### **1. If V incorrectly displays the file as an EBCDIC Carriage Return Delimited file**

In this case, you will be able to see most of the data, but the lines will not line up correctly. To display as RECFM=U:

- Select EBCDIC Options from the View menu (or click on EBC on the status bar)
- Disable the No Formatting option
- Select RECFM=U

##### **2. If the file is displayed as ASCII text or hex**

In this case, the data will be unrecognizable.

To display as RECFM=U, you will first need to enable EBCDIC mode by selecting EBCDIC from the View menu (or pressing Alt+B). Then perform the operations above.

**Note:** **V** will display an error message if it does not recognize the file as RECFM=U.

#### 4.3.174.10.4 Carriage Control

Carriage control is used by some EBCDIC files to indicate line and page breaks.

In files that have carriage control, the first character in each line is used to indicate whether a line or page break should be placed after (or even before) the line.

Carriage control (sometimes referred to as **CCTYPE**) can be one of 3 types:

- ANSI (CCTYPE=A)
- Machine (CCTYPE=M)
- ASCII (CCTYPE=Z)

**V** does not automatically recognize files with carriage control. If your EBCDIC file contains carriage control, you need to select EBCDIC Options from the View menu (or click on **EBC** on the status bar) and select the correct CCTYPE.

If a file contains page breaks, **V** will [paginate](#) it. In particular, **V** will display a page marker (dotted line) before the start of each page.

#### Notes

- Carriage Control is only supported for EBCDIC files - not for ASCII files.

#### 4.3.174.1 Preferences / Configuration Options

The Preferences Dialog Box is where you configure most of the program options. You may select it from the toolbar, the **View** menu, or from various right-click menus.

The tabbed dialog box consists of:

[File Options](#)  
[More Options](#)  
[Window Layout](#)  
[EBCDIC](#)  
[Editor/CMD](#)  
[Fonts](#)  
[Keyboard](#)  
[Line Numbers](#)  
[Search](#)

These are described in the following sections.

#### Note

You must press the **OK** button in order to save the settings that were modified in any of the tabbed dialog boxes. If you press the **Cancel** button, all the modifications will be lost.

#### 4.3.174.11.1 File Options

##### **Tab Size**

Specifies the tab length. If you don't want V to expand tabs, leave this field blank (or set it to 1).

##### **Date Format**

Specifies what format the date will be displayed throughout the program (including when printing).

##### **Always Open as Text**

When V opens a file, it tries to determine what kind of file it is. If it is a binary file (like a **JPG** or **EXE** file) it will display the file in HEX mode, otherwise it will open it in TEXT mode. Check this option if you always want the file to be opened in TEXT mode.

##### **Start at End of File**

If this is checked, V will start viewing from the end of the file instead of the beginning.

##### **Wrap Lines to Screen**

Usually, when displaying files in Text mode, V will not wrap any long lines - you will have to use the horizontal scroll bar to view lines longer than the width of the window. Check this option if you want V to wrap the lines to the width of the window. In this case there will be no horizontal scroll bar.

##### **Wrap on Word Boundary**

Usually when lines are wrapped they are wrapped at the exact position where the line would exceed the width of the window - even if it means splitting the line in the middle of a word. When this option is checked, the line is always wrapped at the end of a word.

##### **Unmark block after copy**

By enabling this option, any highlighted text will be cleared once it is [copied to the clipboard](#) (or saved to a file).

##### **Do not restore file position**

When V views a file that it has viewed before, it will restore the [previous file position](#). Enable this option if you want V to always start viewing files from the beginning.

##### **Hex offset in status bar**

This causes the hex offset to be displayed in the status bar whenever the user clicks the mouse on a file position, while viewing a file in text mode. The hex code of the corresponding character is also displayed (in text mode only).

##### **Display ALL hex codes**

Replaces any dots (unprintable characters) on the right hand side of the [hex dump](#) with the corresponding symbol in the selected font.

##### **Auto-sum columns**

Enable this option if you want V to [automatically display the sum when selecting columns](#).

**Use Bold Cursor**

V will use a **bold** cross-hair cursor when viewing a file.

**Save File Tailing state**

V always disables [File Tailing](#) on startup. Enable this option if you want the previous File Tailing state restored.

**Save Bookmarks on Exit**

By default, all bookmarks are cleared when V exits. Enable this option if you want the bookmarks saved so they will be available every time you view the file.

**Closing File Window to behave like pressing ESC key**

While viewing a file, pressing the upper right [x] button on the window causes V to exit. By enabling this option, V will now treat the [x] button like the ESC key. In particular, if you enable the "Browse in File View" option in the [Keyboard tab](#) of Preferences, V will return to the directory listing when pressing [x] instead of exiting.

**Do not scroll current line marker when using scroll bars**

If the window is scrolled using the scroll bars, the current line marker is moved so that it remains visible. Enable this option if you prefer the current line marker to stay where it is.

**Always scroll window when using arrow keys**

Pressing the Up/Down arrow keys moves the current line marker. The window is only scrolled when the line marker reaches the bottom (or top) of the window. Enable this option if you want the window to always scroll when you press the arrow keys.

**Disable Page Down on Middle Mouse Button**

While viewing a file, V treats the middle mouse button as a Page Down key. This can interfere with the behavior of some mice (like the IntelliMouse) that can use the middle button (or scroll wheel) for panning. You can disable the default V behavior by enabling this option.

**Prompt before reloading modified file**

V will automatically reload the file it is currently viewing if it has been modified by another program. Check this option if you want V to warn you before it does this.

**Do not guess EBCDIC fixed record length (RECFM=F)**

When V detects an [EBCDIC file](#), it tries to determine if it contains fixed length records. However, it can sometimes incorrectly guess the record length. By enabling this option, V will not try to guess the format. Once the file is displayed, the user will have to select the [EBCDIC Options](#) and manually select the correct format.

**Do not copy CR/LF to clipboard for wrapped (to screen) lines**

When wrapping lines to the screen width, V will include a newline (CR/LF) at the screen wrap position whenever the lines are copied to the clipboard. Enable this option if you do not want to include a CR/LF at the screen wrap position. That is, a CR/LF will only be included at the very end of the line.

#### 4.3.174.11.2 More Options

##### **Enable File Chunks**

This indicates that **V** will display large files [a chunk at a time](#), instead of reading the entire file. This greatly reduces the time taken for **V** to load a file.

##### **Chunk Size**

This is the size of the file chunk (in Kb). When File Chunks are enabled, **V** will only load this much of the file, regardless of how large the file is. The default chunk size is 8192 (ie, 8 Mb). [Click here for more details on File Chunks](#).

##### **Enable Smooth Scrolling**

Enables "smooth" scrolling. Note that this is different from the "smooth scrolling" that programs like Internet Explorer support.

##### **Smooth Delay**

Typically a number between 50 and 200 (depending on the speed of your system). [Click here for further details on Smooth Scrolling](#).

##### **Enable MRU File List**

Enable this option if you want **V** to keep a history of the Most Recently Used (MRU) viewed files. These files will be listed on the Recent Files menu. You may also specify the number of files to remember (up to 50).

##### **Word Sets**

You may define your own wordsets which determine what is highlighted when you double-click and shift-double-click on a word. [Click here for further details](#).

## 4.3.174.11.3 Window Layout

**Use Existing File window to view new file  
(unless SHIFT pressed)**

By default, every file you view will be displayed in a separate window. Enable this option if you prefer to only have one file window open at a time. If you enable this option, you can still view files in a separate window by pressing the SHIFT key when you view the file.

**Automatically Tile multiple file windows  
(Auto-Arrange, Vertical, Horizontal)**

Enable this option if you want multiple file windows to be [automatically tiled](#).

**Auto Arrange Grid Size**

This defines the grid size used for [auto-arranging](#).

#### 4.3.174.11.4 EBCDIC

This allows the mapping of the 256 EBCDIC characters to their corresponding ASCII equivalent to be modified. This is necessary because not all EBCDIC character mappings are the same - particularly when it comes to special or control characters.

The current character mapping is displayed in the **EBCDIC to ASCII** table. The first column of the table displays the EBCDIC code, the second displays the ASCII character which it maps to. The code in the second column is also displayed in the **Current Mapping** list box. For example, the EBCDIC character **F0h** represents a 0 (zero) which is **30h** in ASCII. To modify the ASCII character to which the EBCDIC character will be mapped, simply select the new character from the **Current Mapping** list box.

If you wish, the mapping table may be sorted on the ASCII code, by selecting the **ASCII to EBCDIC** option. In this case, the first column of the table will contain the ASCII code and the second column will contain the EBCDIC character which maps to it.

Character codes can be displayed in decimal instead of hex by selecting the **Display as Hex** checkbox.

The **Default Mapping** button can be used to restore the EBCDIC to ASCII mapping to the default.

#### Notes

- If modified, the EBCDIC to ASCII mapping is not saved unless V has been registered.
- The EBCDIC end-of-line character should be mapped to a Line Feed (LF = ASCII 0Ah) and not a Carriage Return (CR = ASCII 0Dh).

#### 4.3.174.11.5 Editor/CMD

**V** is not an editor. If while you are viewing a file, you decide you want to edit it, you must do so with a separate editor. Most people will have an editor of choice on their system. If you don't, you can always use **notepad.exe** which should be in your WINDOWS system directory.

The Editor options tell **V** what editor to use to edit the file. To launch the file in your editor of choice, you need to select the File->Edit command. There are 2 options you can specify - just click on the corresponding button to set them.

**Path** The full path name of your editor (usually an EXE file)

**Options** The options you want passed to your editor (if any)

The above path and options will be used to construct a command that Windows will execute. The command will look as follows:

#### **Path + Options + FileName**

For example,

**\bin\editor.exe /v file.cpp**

Where more than one file has been selected, the file names will be appended onto the end of the command, separated by spaces.

#### **Advanced Options**

You can pass further options (like the current line number) by using [option specifiers](#) (like **%L**). An explanation of the available options specifiers is given in the [User Commands section](#).

#### **Hex Editor**

If you want a different editor to be called while you are viewing a file in hex mode, simply define a second editor in the "Hex Editor" section.

#### **Passing the MSDOS File Name**

Enable the **Use MSDOS file name** option in each of the above cases in order to pass the MSDOS (8.3) file name to the command. This may be necessary if the specified executable is a 16-bit program which does not support long file names.

#### **Command Processor**

You can use a different Command Processor (to TCC.EXE) to launch a Command Prompt by enabling "Use the following Command Processor" and clicking on the Path button to select its path.

#### **Notes**

If you want to use more than one editor, you can define it as a [User Command](#).

---

By default, you can use **Ctrl+E** to launch the editor and **Ctrl+Shift+E** to launch the hex editor.

#### 4.3.174.11.6 Fonts

To change the fonts, simply click on the corresponding button.

##### **Screen Font**

The font used in the File View to display the contents of the file.

##### **Printer Font**

Specifies the font to be used when printing.

##### **2UP Font**

Specifies the font to be used when printing in 2UP mode.

##### **End of Line Indicator**

You can also specify which character is to be used as the **End Of Line Indicator**. This character will be displayed at the end of every line if the [Show EOL](#) option is set, or will be printed at the end of every line if the [Print End of Line](#) option is selected in the Print dialog box.

##### **Use DOS/OEM character set when viewing file**

This is equivalent to the [DOS/OEM Char Set](#) option (on the View menu while viewing a file).

##### **Use DOS/OEM character set when printing**

The same as above - but for printing. Enable this option if you are printing files that contain line drawing characters. Note that this will only work if the selected font supports the OEM character set.

#### **Notes**

- V only allows use of non-proportional (or fixed-pitch) fonts (like Courier) when displaying files. This makes file display, block highlighting and line wrapping much quicker than would be possible using proportional fonts (like Arial or MS Sans Serif).
- Screen fonts can also be selected from the [Fonts menu](#).
- When printing, it is usually best to use the printer's own built in font (if it has any) - it looks much better than Courier. For HP LaserJet printers, you should be able to select a font called LinePrinter. Printer fonts have a small printer icon next to them in the Font Selection List.
- Proportional fonts may be chosen when printing, however, line wrapping will not work correctly. [Click here for further details](#).
- Not all fonts share the same character set. Because of this, you may need to reselect the End Of Line Indicator whenever you select a new font.
- The Printer fonts may also be selected from the [Print Dialog box](#).

#### 4.3.174.11.7 Keyboard

##### **Page Up/Down to go to start of page if file is paginated**

If a file is paginated, enabling this option will cause Page Up/Down to scroll to the start of a new page instead of scrolling the length of the window. [Click here for further details on paginated files.](#)

##### **Escape Key**

The following options determine how the Escape key is treated. Note that some of these options are mutually exclusive.

##### **Close File Window if pressed in the File View**

Pressing ESC will simply close the file window. If another file window is open, **V** will give it focus, otherwise, **V** will return to the Directory Listing.

##### **View Next File**

If multiple files were specified on the Command Line, pressing ESC will view the next file. This is equivalent to pressing Ctrl+Shift+PgDn

#### 4.3.174.11.8 Line Numbers

**V** gives you several options when it comes to displaying/printing line numbers alongside the file data.

##### **Display Line Numbers**

Check this option if you want **V** to prefix each line displayed with its corresponding line number. Line numbers always begin at 1.

##### **Display Line Number 1**

When an increment is specified, the first line in the file (line 1) will usually not be numbered. Check this option if you want the line number displayed on the first line.

##### **Line Number Increment**

This tells **V** how often to display line numbers. For example, if the increment was 10, the line number would be displayed every 10 lines (lines 10, 20, 30, ...). If the increment is left blank (or set to 1), every line will be numbered.

##### **Pad with zero (instead of space)**

The line numbers are always a fixed length, depending on the size of the file. For example, if the file has 199 lines, the line numbers would contain 3 digits. Usually **V** will pad unused digits of the line numbers with spaces. Check this option if you want zeroes used instead. In this case **V** will display "001" instead of " 1".

##### **Reset Line Numbers on New Page (if paginated)**

If the file is [paginated](#), enabling this option will reset the line number to 1 at the start of each page.

##### **Include line numbers on Copy to Clipboard**

Enable this option if you also want the line number(s) included when you copy selected text to the clipboard.

##### **Printing Line Numbers**

The above options also apply when printing a file, however they must be specified separately as they may differ from the display options. For example, you may want to display line numbers but not print them.

The option to enable line numbers when printing is not found in this dialog box, but on the main [Print Dialog Box](#).

Two extra line number options exist when printing files.

**Print First Line in Page**

Check this to print the line number for the **first** line in each page.

**Print Last Line in Page**

Check this to print the line number for the **last** line in each page.

**Note**

These options only apply to **Text** mode. In Hex mode, the hex address is always displayed/printed.

#### 4.3.174.11.9 Search

The options presented here are the same as those presented when you press the Options button in the [Search Dialog box](#).

#### 4.3.174.1 Favorites

Favorites provide an easy way to bookmark frequently viewed files and directories for faster retrieval. Once saved, a favorite may be viewed by selecting it from the Favorites menu.

The currently viewed file or directory may be added to the Favorites by selecting the [Add to Favorites](#) option from the Favorites menu.

When adding a favorite, you can give it a meaningful description which will be displayed in the Favorites menu. If you do not, the path name will be displayed.

You can also store the favorite in a "Folder" by highlighting a folder name in the "Create in" list. You can create a new folder by pressing the "New Folder" button. Favorites stored in folders will appear in popup menus off the main Favorites menu.

The favorites may be modified by selecting the [Organize Favorites option](#).

#### Note

Unregistered versions will only be able to select the first 3 favorites.

4.3.174.12.1 Add To Favorites

### **Adding a File**

You will usually add a file to the Favorites while you are viewing it. When adding a file to the Favorites, you may set the following option:

#### **Restore File Position**

If this is enabled, the current file position will be restored whenever the favorite is selected. Otherwise, the file will be viewed from the start.

#### **Executing a Favorite**

You can also add a file to the Favorites from the Directory View by highlighting it and selecting "Add Selection to Favorites" from the Favorites menu. You will then have the option of enabling the **Execute** option. If this is enabled, the file will be "executed" when selected from the Favorites menu. Otherwise, it will be displayed.

For example, if you add a "Word Document" to your Favorites, enabling the "execute" option will cause the file to be loaded in Word when you select it from the Favorites menu (instead of being displayed by V).

#### 4.3.174.12.2 Organizing Favorites

You may add, delete, move and edit favorites by selecting "Organize Favorites" from the Favorites menu. A favorite may be moved to a new position by simply dragging it and dropping it into its new position. The favorite will be placed **before** the entry on which it was dropped.

Favorites may be created by pressing the "Insert File" or "Insert Directory" buttons. "Insert Copy" will create a copy of the currently highlighted favorite. "Insert Separator" will insert a separator into the Favorites menu.

#### Organizing into Submenus

If you have many favorites, you will probably find it useful to organize them into submenus. To create a new submenu, press the "Insert Submenu" button.

To move a favorite into an empty submenu, simply drop it onto the submenu name. If the submenu is not empty, it will expand so you can drop the favorite into the required position. The dropped favorite will be placed \*before\* the entry it is dropped on. If you want to place the favorite at the \*end\* of the submenu, drop it onto the submenu name you want it placed under.

If you want to move a favorite so that it is positioned just before a submenu, you need to press the SHIFT key as you drop the favorite onto the submenu you want it to precede. If you do not press the SHIFT key, the favorite will be placed inside the submenu.

For further details on modifying favorites, see the following:

- [Favorite Files](#)
- [Sorting Favorites](#)
- [Using Numeric Drive Letters in path names](#)

Pressing the **More** button displays a menu that allows you to [sort](#), [export](#) and [import](#).

#### **Note**

You cannot create a submenu within a submenu.

#### 4.3.174.12.3 Favorite Files

The following may be specified for a favorite file:

##### **File Path**

The file name. Press the "..." button to browse. You can use [numeric drive letters](#) in the file path which will be expanded depending on the environment in which **V** is being run. [Click here for further details.](#)

##### **Tail**

If this is enabled, the file position is set to the end of the file.

##### **Hex**

The favorite is viewed in Hex mode.

##### **EBCDIC**

The favorite is viewed in EBCDIC mode.

##### **Restore File Position**

If this is enabled, the file position will be restored. Otherwise, the file will be viewed from the start.

The file position consists of:

##### **Line Number**

The line number at which to position the file. If the file is to be opened in Hex mode, this will refer to a Hex **offset** instead of a line number.

##### **Column**

The column position.

##### **Chunk**

The chunk to load (if the file is large enough to be loaded in chunks)

##### **Shortcut Key**

You can assign a keyboard shortcut to this Favorite so that it is executed every time the keyboard shortcut is entered. Simply click in the *Shortcut Key* box and enter the desired key combination. Press the ESCape key to clear the shortcut key. A beep will sound if the shortcut key is currently assigned.

##### **Blank File name**

The file name may be left blank. In this case, only the file position and/or mode will be modified.

#### 4.3.174.12.4 Sorting Favorites

Pressing the More button and selecting the *Sort* option will display a dialog box, allowing you to sort the Favorites.

By default, the Favorites will be sorted alphabetically on their description, regardless of whether they are a file, a directory or a submenu. The following options can be set to modify the default behavior.

##### **Place Files before Directories**

If this is enabled, all files will be placed at the top of the list.

##### **Place Directories before Files**

Enable this to place all directories at the top of the list.

##### **Place Submenus at top**

Enable this, to place all submenus at the top.

##### **Place Submenus at bottom**

Enable this, to place all submenus at the bottom.

##### **Sort Submenu contents**

By default, submenu contents will not be sorted. Enable this option to also sort submenus.

#### 4.3.174.12.5 Exporting/Importing Favorites

Pressing the More button and selecting the *Export* option will display a dialog box allowing the Favorites to be exported to a file.

The exported file can then be imported by another user by selecting the *Import* option and specifying the imported file.

##### **Note**

[User Commands](#) and [GridLines](#) can also be exported/imported in the same way as Favorites.

#### 4.3.174.12.6 Using Numeric Drive Letters in Paths

When specifying a file or directory path (in Favorites and User Commands), certain numeric drive letters can be used to specify various system paths. This is particularly useful when using the U3 version of **V** - where the actual file paths can be different every time **V** is run.

The following drive letters are currently defined:

- 0: The folder from where **V** is being executed. On a U3 system, this will be a temporary folder on the host machine, not on the U3 drive.
- 1: The Windows folder (usually C:\Windows)
- 2: The System folder (usually C:\Windows\System32)
- 3: The U3 Drive (eg, X:\)
- 4: The U3 Folder where the **V** settings are stored.
- 5: The user's My Documents folder
- 6: The user's Program Files folder
- 7: The user's Profiles folder (usually C:\Documents and Settings)

#### Note

Numeric Drive Letters can also be used when specifying a [text editor and a Command Processor](#).

#### 4.3.174.1:User Commands

A User Command is any program (usually an EXE file) that the user may wish to execute from **V**.

In the Directory View, the currently selected file(s) may be passed to the User Command and in the File View, the currently viewed file may be passed to the command.

This is a great way to extend **V**.

For example, **V** cannot encrypt files. However, if you already have a program which does this, you can define a User Command which will let you encrypt files using **V**.

User Commands are created by selecting [Organize](#) from the UserCommands menu. Once created, User Commands are executed by selecting them from the UserCommands menu.

#### Keyboard Shortcuts

The default User Command is the first in the list and can be executed by pressing Ctrl+U. The most recently executed User Command can also be repeated by pressing Ctrl+Shift+U. Each User Command can also be assigned its own [keyboard shortcut](#).

#### Running as Administrator

Under Vista or Windows 7, the User Command can be run as an Administrator by right-clicking on the command and selecting *Run As Administrator*

#### Debug Mode

If you right-click on the User Command and select *Run in Debug Mode*, the command to be executed will be displayed before it is run. You will then have the option of running the command or cancelling it. This is a good way of making sure that the correct file names are being passed to the command.

#### 4.3.174.13.1 Organizing User Commands

Organizing User Commands is very similar to [Organizing Favorites](#).

You may add, delete, move and edit User Commands by selecting "Organize" from the UserCommands menu. A command may be moved to a new position by simply dragging it and dropping it into its new position.

User Commands may be created by pressing the "Insert Command" button. "Insert Copy" will create a copy of the currently highlighted command. "Insert Separator" will insert a separator into the UserCommands menu.

#### Organizing into Submenus

If you have many User Commands, you will probably find it useful to organize them into submenus. To create a new submenu, press the "Insert Submenu" button.

To move a command into an empty submenu, simply drop it onto the submenu name. If the submenu is not empty, it will expand so you can drop the command into the required position. The dropped command will be placed \*before\* the entry it is dropped on. If you want to place the command at the \*end\* of the submenu, drop it onto the submenu name you want it placed under.

If you want to move a command so that it is positioned just before a submenu, you need to press the SHIFT key as you drop the command onto the submenu you want it to precede. If you do not press the SHIFT key, the command will be placed inside the submenu.

For further details on modifying User Commands, see the following:

- [Specifying User Command Options](#)
- [Using Option Specifiers](#)
- [Using Numeric Drive Letters in path names](#)

Pressing the **More** button displays a menu that allows you to [sort](#), [export and import](#).

**Note:** You cannot create a submenu within a submenu.

#### 4.3.174.13.2 User Command Options

To define a User Command you must specify the following information and options:

##### **Command Path**

The full path name of the command to be executed. Press the "..." button to browse.

%A can also be entered as the command path. In this case, it will be replaced by whatever program is associated with the selected file when the User Command is invoked.

[Numeric drive letters](#) can be used in the file path, which will be expanded depending on the environment in which V is being run. [Click here for further details](#). Pressing the small question mark button will display a list of valid drive letters.

##### **Shortcut Key**

You can assign a keyboard shortcut to this User Command so that it is executed every time the keyboard shortcut is entered. Simply click in the *Shortcut Key* box and enter the desired key combination. Press the ESCape key to clear the shortcut key. A beep will sound if the shortcut key is currently assigned.

##### **Command Options**

The options that will be passed to the command (if any). This will usually look something like **/option1 /option2**. The options may also contain [option specifiers](#) which are expanded when the user command is run.

##### **Options after file name**

Will place the command options after the file name. See the explanation of the Command Format below.

##### **Start in Command Path**

By default, the working directory of the User Command will be the directory currently being viewed or the directory of the current file. By enabling this option, the working directory will be set to the directory that contains the User Command.

##### **Run As Admin**

Enable this (on Vista and Windows 7) to run the command as an Administrator.

##### **Window**

This describes the state of the User Command window when it is executed. It may be either **Normal**, **Minimized**, or **Maximized**.

##### **Do not pass File names**

By default, any selected files (or directories) will be passed to the command. Enabling this option will cause nothing to be passed to the command (apart from the Command Options).

##### **Prompt for extra options**

By enabling this, the user will be prompted for extra options that will be passed to the command. These options will be appended to any options in **Command Options**.

##### **Use MSDOS names**

If any files are selected, the MSDOS (8.3) form of the file name will be passed to the command.

**Do not allow multiple files**

Enabling this option will ensure that the command is not executed when more than one file is selected.

**Execute command for each file**

If multiple files are selected, all the file names will be passed to the User Command, and the command will be executed once. By enabling this option, the User Command will be executed for each selected file.

**Wait for command to finish**

When executing a User Command, **V** simply launches it and then gets back to business - it does not wait for the command to terminate. In the above case, executing a User Command for each selected file can result in multiple instances of the same program being active at the same time. By enabling this option, **V** will only execute the User Command on a file once the command on the previous file has finished.

**Debug Mode**

If this option is enabled, the User Command will be displayed, and the user asked to confirm if it is to be executed. This allows the user to experiment with [option specifiers](#) without actually having to execute any commands.

**Default User Command**

The **default** user command may be executed by pressing Ctrl-U. The default user command is considered to be the **first** command in the User Command list.

**Command Format**

By default, the actual command that **V** will execute will look as follows:

[Command Path] [Options] [Extra Options] file(s)

If "Options after file name" is enabled the command will look as follows:

[Command Path] file(s) [Options] [Extra Options]

In the case where more than one file name is selected, all the file names are included on the command line, separated by spaces.

**Notes**

- Unregistered versions will only be able to execute the first defined User Command.
- If a [option specifier](#) is used in the Command Options, any selected file names are not automatically added to the command line. If the user wants the file name(s) passed to the command, the appropriate file name specifier (%f or %F) needs to be used.

#### 4.3.174.13.3 Option Specifiers

User commands are constructed by appending the selected file name(s) to the command options.

Although there is an option to place the options last, no further flexibility is available. For example, you cannot place some options before the file name(s) and some after.

Unless you use Option Specifiers.

Option specifiers are entered in the **Command Options** and are expanded when the command is executed. An option specifier consists of a percent sign (%) followed by a single character. The valid specifiers are as follows (note that case is important):

|               |                                                                                     |
|---------------|-------------------------------------------------------------------------------------|
| <b>F</b>      | The selected file name(s) - includes fully qualified path                           |
| <b>f</b>      | File name only (no path)                                                            |
| <b>G</b>      | The fully qualified selected file names(s) with any extension omitted               |
| <b>g</b>      | File name only with extension omitted                                               |
| <b>D</b>      | The name of the current directory - includes fully qualified path                   |
| <b>d</b>      | Directory name only                                                                 |
| <b>Z</b>      | The name of the ZIP file - fully qualified path (Zip View only)                     |
| <b>z</b>      | ZIP file name only                                                                  |
| <b>T or t</b> | The currently selected text (File View or Search Results View)                      |
| <b>U or u</b> | The currently selected (or right-clicked) URL                                       |
| <b>W</b>      | The currently selected (or right-clicked) word ( <a href="#">using word set 1</a> ) |
| <b>w</b>      | using word set 2                                                                    |
| <b>X</b>      | The column number of the start of the selected text                                 |
| <b>x</b>      | The column number of the end of the selected text                                   |
| <b>Y</b>      | The line number of the start of the selected text                                   |
| <b>y</b>      | The line number of the end of the selected text                                     |
| <b>L or I</b> | The line number at the top of the display                                           |
| <b>n</b>      | The number of characters highlighted (in decimal)                                   |
| <b>N</b>      | (in hex)                                                                            |
| <b>s</b>      | The start offset of the selected text (in decimal)                                  |
| <b>S</b>      | (in hex)                                                                            |
| <b>e</b>      | The end offset of the selected text (in decimal)                                    |
| <b>E</b>      | (in hex)                                                                            |
| <b>P or p</b> | Prompts the user for "extra options" which will be appended to the Command Options  |

### Note

A User Command will not be executed if a option specifier cannot be expanded. For example, if **%W** is specified and no text has been selected.

#### 4.3.174.13.4 Sorting User Commands

Pressing the More button and selecting *Sort* will display a dialog box, allowing you to sort the User Commands.

By default, the User Commands will be sorted alphabetically on their description. The following options can be set to modify the default behavior.

##### **Place Submenus at top**

Enable this, to place all submenus at the top.

##### **Place Submenus at bottom**

Enable this, to place all submenus at the bottom.

##### **Sort Submenu contents**

By default, submenu contents will not be sorted. Enable this option to also sort submenus.

#### 4.3.174.13.5 Exporting/Importing User Command

Pressing the More button and selecting the *Export* option will display a dialog box allowing the User Commands to be exported to a file.

The exported file can then be imported by another user by selecting the *Import* option and specifying the imported file.

#### 4.3.174.13.6 Using Numeric Drive Letters in Paths

When specifying a file or directory path (in Favorites and User Commands), certain numeric drive letters can be used to specify various system paths. This is particularly useful when using the U3 version of **V** - where the actual file paths can be different every time **V** is run.

The following drive letters are currently defined:

- 0: The folder from where **V** is being executed. On a U3 system, this will be a temporary folder on the host machine, not on the U3 drive.
- 1: The Windows folder (usually C:\Windows)
- 2: The System folder (usually C:\Windows\System32)
- 3: The U3 Drive (eg, X:\)
- 4: The U3 Folder where the **V** settings are stored.
- 5: The user's My Documents folder
- 6: The user's Program Files folder
- 7: The user's Profiles folder (usually C:\Documents and Settings)

##### **Note**

Numeric Drive Letters can also be used when specifying a [text editor and a Command Processor](#).

#### 4.3.174.1 Printing Files



When the Print command is selected, the Print Dialog Box will appear (which is different from the standard Windows Print Dialog Box). The following options may be specified:

##### **Printer**

The name of the printer to send the file to. All available printers will be listed in the drop-down list box.

##### **Header & Footer**

Check the appropriate box to print a header and/or footer. The text for the header/footer is entered in the corresponding edit box. A history of the previous 10 headers is maintained making it easy to select commonly used headers. [Click here](#) for a description of the **Headers and Footers** formats. If a header/footer is enabled, but no text entered in the edit box, the [default](#) header/footer is printed.

**Override Page Length**

This option is used if you want your page size to have a certain length (in lines). [Click here](#) for further explanation.

**Copies**

The number of copies that you want printed.

**Printer Font**

This button will display the font that V will use for printing. [Click here for more details.](#)

**Profile**

This allows you to save all current settings in a Printer Profile or to restore the settings in a profile. [Click here for further details.](#)

**2Up**

Specifies that you want the document printed in [2UP Mode](#).

**Hex Mode**

This option can only be set if the print was initiated from the Directory View and specifies if the file is to be printed in [Hex mode](#). If the print was initiated from the File View, this option would be disabled and would indicate the mode in which the file was being viewed.

**Vertical Hex Mode**

Prints the file in [Vertical Hex Mode](#)

**Greenbar**

Apply [Greenbar](#) to the printed text.

**Print Line Numbers**

Specifies whether line numbers will be printed with the file. Various options regarding the printed line numbers are set in the [Line Numbers](#) tab of the [Preferences](#) Dialog box. Note that the format of the printed line numbers can differ from that of the displayed line numbers.

**Print End of Line**

Enable this option if you want an End Of Line (EOL) indicator printed at the end of every line. The character that is used for the indicator depends on the font used and can be specified in the [Fonts](#) tab of the Preferences dialog box.

**Duplex (long edge) / Duplex (short edge)**

If your printer supports duplexing (double sided printing), you may also specify if you want to enable short/long edge binding. Note that the duplex options are always enabled - even if your printer does not support duplexing.

**Wrap Long Lines**

[Click here for an explanation of line wrapping.](#)

**Form Feeds**

This determines how Form Feeds will be handled. [Click here for an explanation.](#)

**Margins**

Set the size (in inches) of the top, bottom, left and right margins. [Click here](#) for further details.

**Portrait/Landscape**

Specifies if the file is printed in Portrait or Landscape mode. Note that this option is ignored if the file is to be printed in 2UP mode (which is always printed in Landscape).

**Page #**

Select the pages you want printed. [Click here for further details.](#)

**Line #**

Select the lines you want printed. [Click here for further details.](#)

**Sides**

This lets you select if you want all pages printed or just the **odd/even** numbered pages (which makes double sided printing possible).

**Setup**

The **Setup** button is used to configure the selected printer. The **Print Setup** command from the File Menu can be used to configure the default printer.

**More**

[Click here for further details.](#)

**Notes**

It is always a good idea to do a Preview before printing - especially if the printout is going to be large.

Greenbar may not always preview correctly - but it should print correctly.

Unregistered versions of V have the following restrictions:

- Header/Footer history is not saved
- A ruler may not be printed as a header/footer
- A fixed footer is always printed

#### 4.3.174.14.1 Wrapping Lines

If a line is too long to fit on the printed page, you can have it wrap to the beginning of the next line by checking the **Wrap Long Lines** option.

You must then select the type of wrapping from the adjacent list box. This can be one of:

##### **Right Margin**

In this case, the text is wrapped whenever a line reaches the right margin.

##### **Column**

Select this option to have the text wrapped at a specified column position (which you enter in the adjacent box).

##### **New Page**

When this is selected, long lines are wrapped onto a new page. For example, if a page is 100 characters wide, a line of 300 characters will span 3 pages. That is, the first 100 characters will be printed on the first page, the second 100 on the next and the third 100 on the next.

#### **Notes**

- If this option is not enabled, any long lines will be truncated.
- If you select column wrapping and the column length is too long to fit on the page, the lines will be truncated.

---

#### 4.3.174.14.2 Print Range

This lets you specify what part of the file you want printed. You may select one of the following:

**All**

Prints the entire file

**Page #**

Prints the range of pages you specify in the **From** and **To** boxes.

**Line #**

Prints the range of lines you specify in the **From** and **To** boxes.

**Selection**

Prints the currently selected text. If no text is selected, this option will be disabled.

**From Current Page**

Starts printing from the start of the **current page** - which is the line that is currently displayed at the top of the screen. The number of pages to print is specified in **Pages**. If this is left blank, **V** will print to the end of the file.

#### 4.3.174.14.3 Headers and Footers

User defined headers and footers can be printed on every page. Headers (and footers) each consist of 3 sections - **left**, **center** and **right**, which are left justified, centered and right justified, respectively. To specify a header, you enter each of these 3 sections, separated by a semi-colon. That is - "**left;center;right**" (do not include the quotes).

Each of these sections can contain plain text, special format specifiers or can be empty. The format specifiers consist of a percent (%) followed by one character, and are expanded upon printing. The valid specifiers are as follows (note that case is important):

|           |                                               |
|-----------|-----------------------------------------------|
| <b>%f</b> | Name of the current file (name only)          |
| <b>%F</b> | Full Path Name of current file                |
| <b>%d</b> | Current Date                                  |
| <b>%D</b> | Directory Name (ie, %F without the file name) |
| <b>%e</b> | File Date                                     |
| <b>%t</b> | Current Time (24 hour format)                 |
| <b>%T</b> | Current Time (12 hour format)                 |
| <b>%u</b> | File Time (24 hour format)                    |
| <b>%U</b> | File Time (12 hour format)                    |
| <b>%p</b> | Current Page Number                           |
| <b>%P</b> | Total pages to be printed                     |
| <b>%r</b> | Print the Ruler                               |
| <b>%g</b> | Print the Gridlines                           |

If you want to use a "%" or ";" in the header text - prefix them with a "%". That is, use "%%" and "%;" respectively.

Examples: (once again, do not enter the quotes)

**"%f;;Page %** Print the file name on the left and page number on the right  
**d"**  
**";;%d;"** Just print the page number (with no text) in the center  
**";;"** Prints an empty header/footer

If the header/footer field is left blank, it defaults to "**%f;%d %t;Page %p**". That is, it prints the file name on the left, the date and time in the center, and the Page number on the right.

#### Notes

- The ruler and gridlines cannot be combined with any other specifier (only each other). For example, you cannot combine the ruler with a page number. If "%r" is specified, then anything else that may be entered in the header/footer is ignored (with the exception of %g).
- If both a ruler and a grid are specified ("%r %g" or "%g %r"), the grid will always be displayed after the ruler if printed as a header, and before the ruler if printed as a footer.
- The ruler specifier will be ignored unless **V** has been registered.

#### 4.3.174.14.4 Form Feeds

Form feed characters (ASCII 12 or Ctrl-L) are generally used in text files to signify a page break.

You may select one of the following 3 Form Feeds options in the Print Dialog Box:

**New Page** V will start a new page every time a form feed is encountered.

**Draw Line** A page separator (dotted line) will be printed whenever a Form Feed is encountered - a new page will not be started.

**Ignore** The form feed will be treated as a normal character - and will be printed. The appearance of the printed form feed will depend on the print font.

#### 4.3.174.14.5 Margins

The margins specify the distance from the text to the edge of the page in each direction and are always specified in inches. For those who are only familiar with centimetres, **1 inch** is equal to **2.54cm**.

Modifying the size of the margins affects the size of the page that is available to print the file - the larger the margins, the smaller the area available to print the file.

#### 4.3.174.14.6 Page Length

Usually, the number of lines that can fit on a page is determined by the physical length of the page, the size of the margins and the height of the printer font.

At times, files are pre-formatted to a particular page length (usually around 60 lines). That is, the file usually contains its own header and/or footer every 60 lines. Printing a file which has been pre-formatted to 60 lines on a page that is physically 66 lines long will look awkward - headers and footers will start appearing all over the pages, instead of where they should be!

To overcome this problem, you can override the physical page length by specifying the length of the printer page. This causes V to start a new page as soon as the specified number of lines has been printed instead of waiting until the end of the page.

#### Notes

- The page length specified must be less than or equal to the maximum page length allowed by the printer.
- Pre-formatted files usually contain their own header/footer, so you will probably have to disable V printing any of its own.
- If your listing is already formatted to a certain page length which is larger than your printer page length, you will have to increase the size of the printer page (by reducing the top and bottom margins and/or disabling the header/footer) or reduce the size of the printer font.
- Some files contain form feed characters (Ctrl-L or ASCII 12 decimal) to indicate a page break. V will start a new page whenever it encounters a form feed.

#### 4.3.174.14.7 2UP Printing

2UP printing would probably have to be one of the most useful (and most used) features of **V**. At least it is for me!

2UP printing not only saves paper, but I find the listings actually look better and are easier to read, since you have more information on the one page.

When files are printed in 2UP mode (also known as book mode), the file is printed in Landscape mode with two pages being printed (side by side) on each sheet of paper.

2UP printing is ideal for program listings, hex dumps and README files.

#### Notes

- When printing in 2UP mode, you should use a smaller font than you would use for normal printing. On a HP LaserJet, the built-in LinePrinter font is ideal.
- See the [Fonts](#) section of the [Preferences](#) Dialog box for further information on selecting Printer fonts.
- The Orientation option (Portrait/Landscape) is ignored when 2UP printing is selected - the printer is always placed in Landscape mode.

#### 4.3.174.14.8 Printer Fonts

The font that **V** will use for printing will be displayed in a button just above the "Options" group in the dialog box. You may change the font by clicking on the button and selecting a previously used font from the list displayed. You may add a font to the list by selecting **Add Font**.

Select **Organize Fonts** if you want to modify the font list.

When you change printing modes (Hex/Text and 2Up/Normal) **V** will automatically select the font last used in that mode.

#### Proportional fonts

Proportional fonts may not be selected for displaying files but they may be selected for printing. Proportional fonts will not work well for program listings and hex dumps since the spacing between characters is not fixed (it is proportional). However, proportional fonts may be preferable for printing text files.

#### Line Wrapping

Line wrapping will not work correctly if a non-proportional printer font is selected. In particular, the lines will usually wrap well before the end of the page.

If proportional fonts are used, it is suggested that the **Wrap Long Lines** option not be set.

#### **Note**

Selecting **Fonts** from the menu on the [More button](#) allows the user to select a new printer font. This is equivalent to selecting fonts from the [Preferences](#) dialog box. However, the font selected will not be added to the font list which is displayed when clicking on the button displaying the current print font.

#### 4.3.174.14.9 More Printing Commands

Pressing the **More** button allows you to select one of the following:

##### **Fonts**

This allows the user to select a new printer font. The font selected will not be added to the font list that is displayed when pressing the button displaying the current font.

##### **Text Only Options**

Sets the options for [Text Only Printing](#)

##### **Start Text Only Printing**

Starts [Text Only Printing](#)

##### **Start Raw/Binary Printing**

Starts [Raw/Binary Printing](#)

##### **Apply Settings & Exit**

This saves the print options and closes the Print dialog box. Note that pressing the Cancel button will not save any options that have been modified.

##### **Note**

You can start Text Only and Binary printing from the command line by specifying the **/PX** and **/PB** command line options respectively.

#### 4.3.174.14.10 Text Only Printing

**Text Only** printing causes V to send text directly to the printer, bypassing the Windows printer driver. This will normally be used to print to a very old printer that is not supported by Windows.

There are 2 ways of starting Text Only printing.

The first is to simply select **Start Text Only Printing** from the menu that is displayed when you click on the [More button](#).

The second is to print to a printer that uses the **Generic/Text Only** printer driver. This is usually found under the manufacturer of **Generic** in the Add Printer Wizard. If a generic printer driver is used, V will bypass the Windows driver and print directly to the printer (unless Text Only printing has been disabled).

Text Only printing may be configured by selecting **Text Only Options** from the menu that is displayed when the [More button](#) is pressed. You will probably need to know certain technical information about the printer in order to configure it correctly (do you still have the manual?). The following options may be specified:

##### **Disable Text Only Printing**

Enable this if you really want to use the Windows printer driver (which will normally be bypassed).

##### **Page Size**

The page size of the printer in columns and rows. The column size will usually be 80 or 132 and the number of lines between 60 and 66.

##### **Margins**

The number of columns (or lines) to skip before printing each page. Note that the Page Size should include the margins. For example, if the page width was 80 columns and the left and right margins were 5 characters, the printable page width would be 70 characters.

##### **Send after each LINE**

This tells V what to send to the printer at the end of a new line. It will usually be one of:

|              |                                                                                                                                            |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <b>CR</b>    | Carriage Return (Hex 0D)                                                                                                                   |
| <b>LF</b>    | Line Feed (Hex 0A)                                                                                                                         |
| <b>CRLF</b>  | A CR followed by a LF                                                                                                                      |
| <b>Other</b> | You may specify and end of line character sequence if it is not one of the above. The format of the character sequence is described below. |

##### **Send after each PAGE**

What to send to the printer at the end of each page to advance to the next page. It will usually be one of:

|                    |                                                                                      |
|--------------------|--------------------------------------------------------------------------------------|
| <b>FF</b>          | Form Feed (Hex 0C)                                                                   |
| <b>Blank Lines</b> | Advances to the next page by a series of blank lines (dependant on the page height). |
| <b>Nothing</b>     | Send nothing                                                                         |
| <b>Other</b>       | Specify your own end of page character sequence.                                     |

**Send at Start of file**

Character sequence to send at the start of the file (perhaps to put the printer in condensed mode).

**Send at END of file**

Character sequence to send at the end of the file.

**Delay after each LINE and/or PAGE**

You may also tell **V** to wait after printing each line and/or page. The delay is specified in milliseconds. For example, 500 milliseconds equals half a second.

A delay may be necessary if you find that the printer is losing characters due to its buffer not being large enough.

**Character Sequence Format**

To specify a character sequence to be sent to the printer, simply type in the characters (if they are alphanumeric). For control characters, type a **%** followed by the **2 digit hex code**.

For example, to send a LF/CR instead of a CR/LF you would specify **%0a%0d**. To send the **ESCAPE** (hex 1b) character followed by the letter **A** you would specify **%1bA**.

**Notes**

See also [Raw/Binary Printing](#)

Print Preview will probably not work correctly for Text Only printing.

#### 4.3.174.14.11 Raw /Binary Printing

Text Only printing lets you send the file directly to the printer, bypassing the Windows printer driver.

This is only really useful for printing \*text\* files, as **V** will try to paginate the files. That is, start a new page depending on how many lines in a page.

Raw/Binary printing, however, does not paginate the file. It sends the file unmodified to the printer.

Raw/Binary printing can be used to print a file that already contains printer control codes, and needs to bypass the printer driver in order to print correctly.

#### 4.3.174.14.12 Printer Profiles

A Printer Profile is a collection of printer settings (header, footer, font, margins, orientation, etc).

To save the current printer settings in a profile, click on the **Profile** button and select Save in New Profile. You will then be asked to enter a profile name. This name will appear in the Profile drop-down list box in the Print dialog box.

When a profile is selected from this list box, all the printer settings stored in the profile will be restored.

Once a profile has been selected, any settings changes will not automatically be saved back to the profile. You will need to do this manually by clicking on the **Profile** button and selecting Save in Current Profile.

When a profile has been modified (without being saved), the Profile list box will display "Profile Name (Modified)" to indicate that the current settings are different from the saved profile. Note that this will not happen as soon as the options are modified - but the next time the Print dialog box is displayed.

#### **Default Profile**

A default profile can be defined by clicking on the **Profile** button and selecting Set as Default Profile.

Defining a default profile allows the user to revert to the default profile settings whenever **V** is started or after a specified number of minutes from the last print. To configure this, click on the **Profile** button and select Default Profile Options.

#### **Notes**

Profiles cannot be used to save [Text Only options](#)

Profiles are stored as **.vprofile** files in the user's Application Data folder - usually in:

C:\Documents and Settings\UserName\Application Data\V\Profiles\

#### 4.3.174.1!Keyboard Shortcuts

Most of the Menu and Toolbar commands in **V** have a keyboard equivalent (and in many cases, more than one).

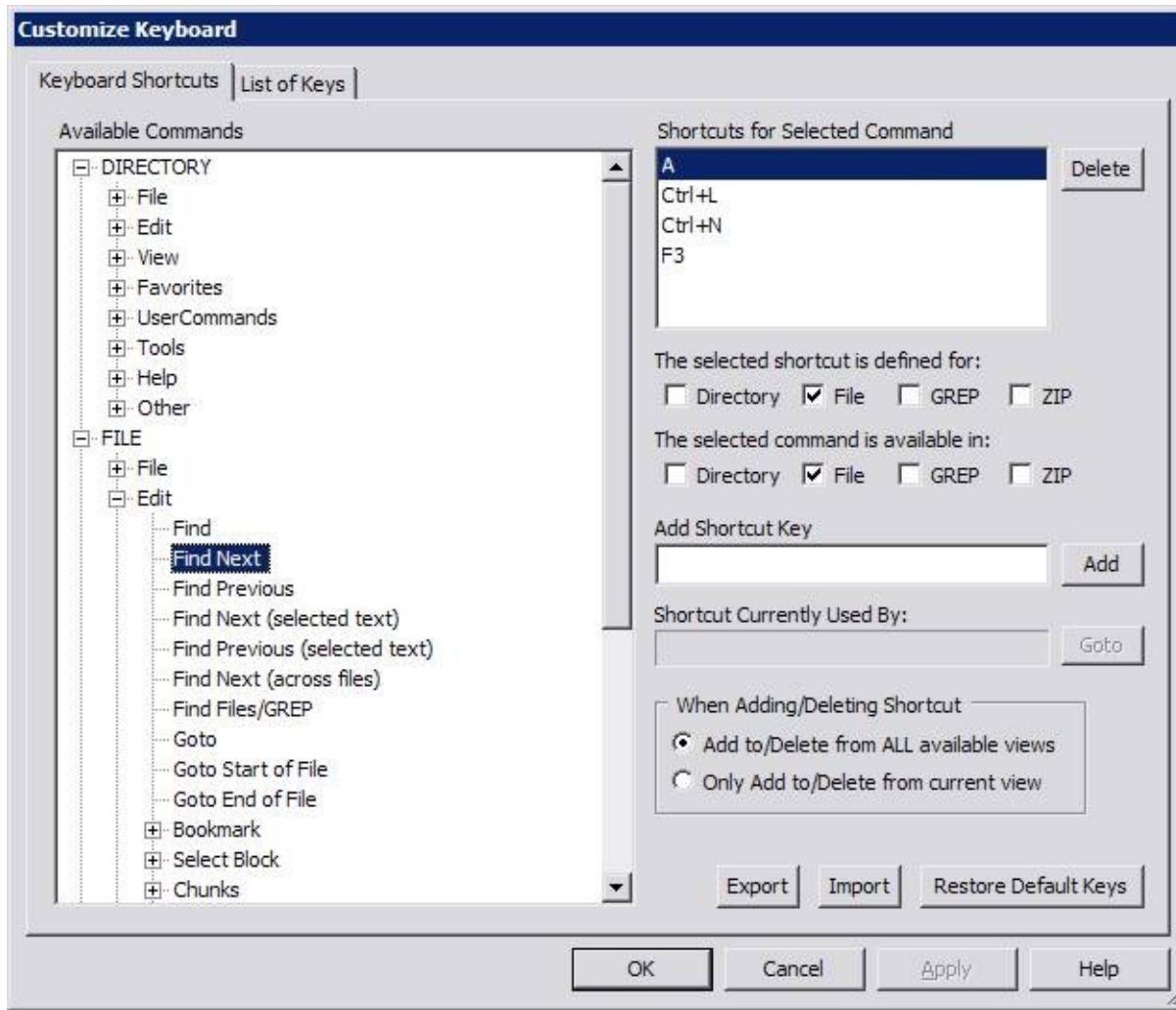
To many, it may seem that a lot of the keys have been selected at random. I'd like to think that there is a bit more to it than that. Many **V** users have been using PCs for a long time and they all have their own preferences when it comes to utilities, and in particular, to editors.

No one wants to learn a new set of keyboard commands. Windows and CUA was supposed to fix all this, but try telling someone who uses **vi** all day (a lot of people still do!) that they have to press "**Ctrl-F**" to search, when they are used to typing "/".

Most of the keyboard shortcuts can be customized. This allows the user to define a new keyboard shortcut or to re-assign an existing one. [Click here for details.](#)

#### 4.3.174.15.1 Customizing the Keys

The keyboard can be customized by selecting *Customize Keyboard* from the Tools menu.



A tree will be displayed that will contain the top level menus in each of V's four views (Directory, File, GREP and ZIP). Commands that do not appear on any menus (like Goto Directory Box in the Directory View and Page Down in the File View) are listed underneath the **Other** branch of the corresponding view..

When you expand the tree and click on of the commands, the shortcut keys currently corresponding to that command (if any) will be displayed underneath *Shortcuts for Selected Command*.

Note that a command can have multiple shortcuts keys assigned to it. A key listed in **bold** (eg, **Alt+0**) indicates that the key cannot be deleted (or re-assigned). However, extra shortcut keys can be defined for that command.

The above screenshot shows the keys currently assigned to Find Next (A, Ctrl+L, Ctrl+N and F3).

The *selected shortcut is used in* indicates that the selected key (A) is currently only defined in the File

view.

The *selected command is available in* indicates that the Find Next command is only available in the File view.

Click on the following for further details:

[Adding/Deleting Keys](#)

[List of Keys](#)

#### 4.3.174.15.2 Adding/Deleting Keys

To delete a shortcut key, simply select the key and press the *Delete* button. To assign a new shortcut to the command, click in the *Add Shortcut Key* box, enter the new key (or key combination) and press the *Add* button. Press the *ESCAPE* key to clear the *Add Shortcut Key* box.

When adding a new key, *Shortcut Currently Used By* will display any existing command that the key is already assigned to. You will have to delete this key from the existing command before you can assign it to the new command. Pressing the *Goto* button will take you to the menu command that is currently used by the key.

When adding or deleting a key, it will be added to (or deleted from) all available views unless *Only Add to/Delete from current view* is enabled.

For example, *Ctrl+Enter* displays the File Properties in the File, Directory and GREP views. By enabling *Only Add to/Delete from current view*, it is possible to only redefine *Ctrl+Enter* in the File View and maintain its existing functionality in the Directory and GREP views.

#### 4.3.174.15.3 Export/Import Keys

Press the *Export* button to export the shortcut keys to a .vkey file. Note that only the keys that have been customized by the user are exported to the file.

Press the *Import* button to import the customized keys from a .vkey file. Any keys that you have customized will remain customized, unless they have been redefined in the .vkey file.

#### Note

The .vkey file should not be edited

#### 4.3.174.15.4 List of Keys

The *List of Keys* tab displays a list of all the keys currently used by V.

| Customize Keyboard |                          |              |      |      |     |       |   |
|--------------------|--------------------------|--------------|------|------|-----|-------|---|
| Keyboard Shortcuts |                          | List of Keys |      |      |     |       |   |
| Key +              | Command                  | Dir          | File | Grep | Zip | Other | ▲ |
| A                  | Find Next                |              | X    |      |     |       |   |
| Shift+A            | Find Previous            |              | X    |      |     |       |   |
| Ctrl+A             | Select Entire File       |              | X    |      |     |       |   |
| Ctrl+A             | Select All Files         | X            |      | X    |     | X     |   |
| <b>Alt+A</b>       | Find Next (across files) |              | X    |      |     |       |   |
| <b>Alt+A</b>       | ASCII Chart              | X            |      |      | X   |       |   |
| <b>Alt+Shift+A</b> | ASCII Chart              | X            | X    |      |     | X     |   |
| B                  | Goto File End            |              | X    |      |     |       |   |
| Ctrl+B             | File Attributes          | X            | X    | X    |     | X     |   |
| <b>Alt+B</b>       | Toggle EBCDIC Mode       |              | X    |      |     |       |   |
| Ctrl+C             | Copy to Clipboard        |              | X    |      |     |       |   |
| <b>Alt+C</b>       | Control-C (Copy)         | X            |      | X    |     |       |   |
| <b>Alt+C</b>       | Command Prompt           | X            | X    | X    |     | X     |   |
| Ctrl+Shift+C       | Copy                     | X            |      | X    |     |       |   |
| Ctrl+Shift+C       | Append Selected Text     |              | X    |      |     |       |   |
| Ctrl+D             | Delete                   | X            | X    | X    |     |       |   |

An X is displayed in a column if that key is available in the corresponding view. The four views are:

- Dir** The Directory View (ie, the Directory Listing)
- File** The File View (viewing the file contents)
- Grep** The GREP View (where V displays the GREP results)
- Zip** The ZIP View (viewing the contents of a ZIP file)

The *Other* column can contain one of the following:

- FAV** The key is assigned to a [Favorite](#)
- UCMD** The key is assigned to a [User Command](#)
- FONT** The key is assigned to a [font](#)
- GRID** The key is assigned to a [grid](#)

You can **delete a key** by right-clicking on the key in the list and selecting *Delete Key(s)*. Note that keys displayed in **bold** cannot be deleted (and therefore, cannot be re-assigned).

### Scroll Keys

If you **press a key** while viewing the list of keys, the list entry corresponding to that key will be selected (if it exists). For example, if you press Ctrl+A, the first list entry for Ctrl+A will be selected. Pressing Ctrl+A again will select the next entry for Ctrl+A (if it exists).

This causes a problem if you use one of the scroll keys to scroll the list (like PageUp/PageDown). Pressing PageUp will select the PageUp key in the list instead of scrolling the list. If you prefer the scroll keys to scroll the list, enable the *Scroll keys for the above list* option.

### Sorting the List of Keys

The list of keys can be "sorted" by clicking on one of the column headers. Sorting on one of the view types (Dir, File, Grep, Zip) will display all of the keys defined in the corresponding view at the top of the list. For example, sorting on **Dir** will display all of the keys defined in the Directory View at the top of the list.

#### **4.3.174.1(C)Copyright**

**VIEW** is a customized version of **The V File Viewer** licensed to JP Software Inc. for use with Take Command.

**V** implements regular expressions using the PCRE library written by Philip Hazel.

The V File Viewer is Copyright © 1996-2013, Charles Prineas, All Rights Reserved.

Take Command is Copyright©1993-2013, Rex Conn and JP Software Inc. All Rights Reserved.

PCRE is Copyright © 1997-2013 University of Cambridge. All Rights Reserved.

#### **4.3.175 VOL**

**Purpose:** Display disk volume label(s)

**Format:** VOL [*d:*] ...

*d:* The drive or drives to search.

**Usage:**

Each disk may have a volume label, created when the disk is formatted or with the external LABEL command. Also, every disk formatted with Windows has a volume serial number.

The VOL command will display the volume label and, if available, the volume serial number of a disk volume. If the disk doesn't have a volume label, VOL will report that it is "unlabeled." If you don't specify a drive, VOL displays information about the current drive:

```
[c:\] vol
Volume in drive C: is MYHARDDISK
```

If available, the volume serial number will appear after the drive label or name.

To display the disk labels for drives A and B

```
[c:\] vol a: b:
Volume in drive A: is unlabeled
Volume in drive B: is BACKUP_2
```

VOL will also return volume information for UNC names.

See also: [@LABEL](#).

#### 4.3.176 VSCRPUT

**Purpose:** Display text vertically in the specified color

**Format:** VSCRPUT row col [BRIght] fg ON [BRIght] bg text

|             |                       |
|-------------|-----------------------|
| <b>row</b>  | Starting row          |
| <b>col</b>  | Starting column       |
| <b>fg</b>   | Foreground text color |
| <b>bg</b>   | Background text color |
| <b>text</b> | The text to display   |

See also: [SCRPUT](#).

**Usage:**

VSCRPUT writes text vertically on the screen rather than horizontally. It can be used for simple graphs and charts generated by batch files.

Like the SCRPUT command, it uses the colors you specify to write the text. See [Colors and Color Names](#) for details about colors and color names, and notes on the use of bright background colors.

The **row** and **column** values are zero-based, so on a 25 line by 80 column window valid rows are 0 - 24 and valid columns are 0 - 79. VSCRPUT checks for a valid **row** and **column**, and displays a "Usage" error message if either value is out of range.

The maximum **row** value is determined by the current height of the **TCC** window. The maximum **column** value is determined by the current virtual screen width (see [Resizing the Take Command Window](#) for more information).

You can also specify the **row** and **column** as offsets from the current cursor position. Begin the value with a plus sign [**+**] to move down the specified number of rows or to the right the specified number of columns before displaying text, or with a minus sign [**-**] to move up or to the left.

If you specify 999 for the **row**, VSCRPUT will center the text vertically. If you specify 999 for the **column**, VSCRPUT will center the text horizontally.

VSCRPUT does not move the cursor when it displays the **text**.

The following batch file fragment displays an X and Y axis and labels them:

```
cls bright white on blue
drawhline 20 10 40 1 bright white on blue
drawvline 2 10 19 1 bright white on blue
```

```
scrput 21 20 bright red on blue X axis
vscrput 8 9 bright red on blue Y axis
```

### 4.3.177 WAKEONLAN

**Purpose:** Sends a "Wake-On\_LAN" packet to a system

**Format:** WAKEONLAN remotehost macaddress

|                   |                                         |
|-------------------|-----------------------------------------|
| <b>remotehost</b> | The IP address of the machine to wake   |
| <b>macaddress</b> | The physical address of the remote host |

**Usage:**

WAKEONLAN send a "Wake-On-LAN" packet to the specified system (which may also be a broadcast address). This will power on the remote machine if the functionality is supported by the network card on the remote machine.

### 4.3.178 WEBFORM

**Purpose:** POST data to interactive web pages or scripts

**Format:** WEBFORM [/An /En /Fn /U"username" /O:headers" /P"password" /R"referer" /Tn /V] /  
W"url" "varname" "varvalue" ...

|                 |               |
|-----------------|---------------|
| <b>varname</b>  | Form variable |
| <b>varvalue</b> | Form value    |

|                                     |                                |
|-------------------------------------|--------------------------------|
| <a href="#">/An (authorization)</a> | <a href="#">/R(ferrer)</a>     |
| <a href="#">/En (encoding)</a>      | <a href="#">/Tn (firewall)</a> |
| <a href="#">/F (HTTP agent)</a>     | <a href="#">/U(ser)</a>        |
| <a href="#">/L(ocal file)</a>       | <a href="#">/V(erbose)</a>     |
| <a href="#">/O(ther headers)</a>    | <a href="#">/W(eb URL)</a>     |
| <a href="#">/Password)</a>          |                                |

**Usage:**

WEBFORM will POST data to interactive web pages or scripts (CGI, ASP, etc.), similar to what HTML forms do.

WEBFORM will use the proxy & firewall settings from TCMD.INI.

Example:

```
webform /v /w"http://download.finance.yahoo.com/d/quotes.csv" "f", "sl1d1t1c1ohgv" "e", ".csv"
"s", "IBM"
```

**Options:**

**/An** Authorization scheme:

- 0 - basic
- 1 - digest
- 2 - proprietary

- 3 - none
- 4 - NTLM
- 5 - Negotiate

**/En** Encoding:

- 0 (URLEncoding) This is the most common encoding for HTML form contents.
- 1 (MultipartFormData) This is MIME encoding allowing transmission of binary data.
- 2 (QueryString) This is an older form of encoding where the actual parameters are appended to the URL query string. (Generally not recommended because most servers limit the size of the URL to less than 1K or 2K).

**/F** Email address of the HTTP agent.

**/U"username"** User name if authentication is to be used.

**/P"password"** Password if authentication is to be used.

**/L"localfile"** Local file for downloading. If the file exists, it will be overwritten.

**/O"headers"** Other headers. The headers must be of the format "header: value" as described in the HTTP specifications. Header lines should be separated by CR/LF (^r^n).

**/R"referer"** The document referring the requested URL

**/Tn** Firewall type:

- 0 - no firewall (default)
- 1 - Connect through a tunneling proxy. Port is set to 80.
- 2 - Connect through a SOCKS4 proxy. Port is set to 1080.
- 3 - Connect through a SOCKS5 proxy. Port is set to 1080.

**/V** Display retrieved document text

**/W"url"** URL of web page

#### 4.3.179 WEBUPLOAD

**Purpose:** Upload files to RFC1867-compliant web servers

**Format:** WEBUPLOAD [/An /En /F"from" /L"file" /O"headers" /U"username" /P"password" /R"referer" /V] [/W"url" [/V "varname" "varvalue"] [/filevar "filename" ...]

- varname** Form variable
- varvalue** Form value
- filevar** The file(s) to extract
- filename** The file(s) to upload

|                                     |                                |
|-------------------------------------|--------------------------------|
| <a href="#">/An (authorization)</a> | <a href="#">/R(eferrer)</a>    |
| <a href="#">/En (encoding)</a>      | <a href="#">/Tn (firewall)</a> |
| <a href="#">/F (HTTP agent)</a>     | <a href="#">/U(ser)</a>        |
| <a href="#">/L(local file)</a>      | <a href="#">/V(erbose)</a>     |

[/O\(ther headers\)](#)  
[/P\(ssword\)](#)

[/W\(eb URL\)](#)

**Usage:**

WEBUPLOAD will use the proxy & firewall settings from TCMD.INI.

**Options:**

**/An** Authorization scheme:

- 0 - basic
- 1 - digest
- 2 - proprietary
- 3 - none
- 4 - NTLM
- 5 - Negotiate

**/En** Encoding:

- 0 (URLEncoding) This is the most common encoding for HTML form contents.
- 1 (MultipartFormData) This is MIME encoding allowing transmission of binary data.
- 2 (QueryString) This is an older form of encoding where the actual parameters are appended to the URL query string. (Generally not recommended because most servers limit the size of the URL to less than 1K or 2K).

**/F"from"** Email address of the HTTP agent.

**/U"username"** User name if authentication is to be used.

**/P"password"** Password if authentication is to be used.

**/L"localfile"** Local file for downloading. If the file exists, it will be overwritten.

**/O"headers"** Other headers. The headers must be of the format "header: value" as described in the HTTP specifications. Header lines should be separated by CR/LF (^r^n).

**/R"referer"** The document referring the requested URL

**/Tn** Firewall type:

- 0 - no firewall (default)
- 1 - Connect through a tunneling proxy. Port is set to 80.
- 2 - Connect through a SOCKS4 proxy. Port is set to 1080.
- 3 - Connect through a SOCKS5 proxy. Port is set to 1080.

**/V** The following two arguments are a varname / varvalue pair.

**/W"url"** URL of web page

### 4.3.180 WHICH

**Purpose:** Display the command type and what it would execute

**Format:** WHICH [/A] command [command ...]

**command** One or more commands or files.

[/A\(I\)](#)

**Usage:**

WHICH displays information about internal and external commands, [Aliases](#) (including keystroke aliases), and files. When a file matches an applicable [Executable Extension](#) or [Windows File Association](#), that data will be displayed. The exact information reported depends on the type of command or file you specify. For example:

```
[c:\] which cdd buildtree notepad test.btm test.exe test.xyz test.doc
donothing
CDD is an internal command
buildtree is an alias : cdd /s
notepad is an external: C:\windows\notepad.exe
test.btm is a batch file : C:\test.btm
test.exe is an external : C:\test.exe
test.xyz is an executable extension : C:\path\mybatch.btm C:\test.xyz
test.doc is associated with : C:\Program Files\Microsoft Office\OFFICE11
\WINWORD.EXE
donothing is an unknown command
```

If the command is an abbreviated alias, WHICH will display the full name; i.e.:

```
[c:\] alias opt*ions=*option
[c:\] which opt
opt*ions is an alias : *option
```

WHICH can also recognize [Plugin](#) commands, [REXX](#) files, [EXTPROC](#) files, and associated files.

**Note:** WHICH does not support wildcard specifications unless you use the [/A](#) option. Parameters must be actual commands or actual file names (including variable and function references as in "which % comspec"). If a filename includes white space or special characters, it must be enclosed in double quotes. A file specified without an explicit path must be on the current [PATH](#).

See [Executable Files and File Searches](#) for details on the order in which various locations are searched.

See also: [@SEARCH](#), [ASSOC](#), [FTYPE](#).

**Option:**

**/A** Display all matching commands. (Normally WHICH only displays the first match.) Executable files will be displayed in the order they are found in the [PATH](#).

### 4.3.181 WINDOW

**Purpose:** Minimize or maximize the current window, restore the default window size, or change the window title

**Format:** WINDOW [ MAX | MIN | RESTORE | HIDE | TRAY | TOPMOST | NOTOPMOST | TOP | BOTTOM | DETACH | /POS=left,top,width,height | /SIZE=rows,columns | /TRANS=n | /FLASH=type,count | "newtitle" ]

|                 |                                           |
|-----------------|-------------------------------------------|
| <i>newtitle</i> | A new title for the window                |
| <i>height</i>   | New height of window                      |
| <i>width</i>    | New width of window                       |
| <i>left</i>     | New position of the left border of window |
| <i>top</i>      | New position of the top border window     |
| <i>rows</i>     | New height of window                      |
| <i>columns</i>  | New width of window                       |
| <i>type</i>     | Type of window flash                      |
| <i>count</i>    | Number of times to flash the window       |

|                             |                                          |
|-----------------------------|------------------------------------------|
| <a href="#">/FLASH</a>      | <a href="#">/SIZE (of screen buffer)</a> |
| <a href="#">/POS(ition)</a> |                                          |

See also: [ACTIVATE](#) and [TITLE](#).

**Usage:**

**WINDOW** is used to control the appearance and title of the current (**TCC**) window.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

If you are running in a **Take Command** tab window, the MAX, MIN, RESTORE, HIDE, TRAY, TOPMOST, NOTOPMOST, TOP, BOTTOM, and /TRANS options will be sent to the main **Take Command** window, not the **TCC** window.

**Note:** You can specify only one **WINDOW** option at a time. The different options cannot be combined in a single **WINDOW** command. To perform multiple operations you must use multiple **WINDOW** commands.

**Options:**

| <b>Option</b>                              | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|--------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| /<br><b>POS=left,top,width,height</b><br>t | Set the window position and size on the desktop. The values are specified in pixels. <b>Left</b> and <b>top</b> refer to the position of the top left corner of the window relative to the top left corner (0,0) of the screen. The <b>width</b> and <b>height</b> values determine the window size. The window may be sized and positioned so that parts of it are beyond the physical area of the display. The / before the keyword is optional. <b>/POS</b> is not supported in <b>Take Command</b> tab windows (use <a href="#">ACTIVATE</a> instead). |
| <b>/TRANS=n</b>                            | Set the transparency level of the <b>Take Command</b> window. <b>n</b> is an value from 0 (invisible) to 255 (opaque).                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>/FLASH=type,count</b>                   | Flash the <b>TCC</b> or <b>Take Command</b> window. (Not available in <b>TCC/LE.</b> ) The arguments are:<br><b>type</b> - type of flash; one or more of the following values:                                                                                                                                                                                                                                                                                                                                                                             |

|                           |                                                                                                                                                                                                                                                                                                                             |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                           | 0 - stop flashing<br>1 - flash the window caption<br>2 - flash the taskbar button<br>4 - flash continuously until WINDOW is called again with the /FLASH type set to 0<br>2 - flash continuously until the window comes to the foreground (cannot be used with 4)<br><b>count</b> - the number of times to flash the window |
| <b>newtitle</b>           | Changes the window title. The title text must be enclosed in double quotes. (The quotes will not appear as part of the actual title as displayed.) Setting the title inside a batch file will only change the window title while the batch file is executing.                                                               |
| <b>MAX</b>                | Expands the window to its maximum size.                                                                                                                                                                                                                                                                                     |
| <b>MIN</b>                | Reduces the window to an icon.                                                                                                                                                                                                                                                                                              |
| <b>RESTORE</b>            | Returns the window to its default size and location.                                                                                                                                                                                                                                                                        |
| <b>HIDE</b>               | Makes the window invisible. Use <b>RESTORE</b> to make the window visible.                                                                                                                                                                                                                                                  |
| <b>TRAY</b>               | Moves the window to the taskbar tray.                                                                                                                                                                                                                                                                                       |
| <b>/SIZE=rows,columns</b> | Specify the <b>TCC</b> screen buffer size. Due to the design of Windows console sessions, you cannot use <b>/SIZE</b> to reduce the size of the screen buffer; it can only be increased. Does not affect window size.                                                                                                       |
| <b>TOPMOST</b>            | Keeps the <b>Take Command</b> window on top of all other windows until it closes, or <b>NOTOPMOST</b> is used. (Only valid in a tab window.)                                                                                                                                                                                |
| <b>NOTOPMOST</b>          | Allows other windows to overlay the <b>Take Command</b> window (this is the normal state for most windows). (Only valid in a tab window.)                                                                                                                                                                                   |
| <b>TOP</b>                | Moves the <b>Take Command</b> window to the top of the window order, above all other non- <b>TOPMOST</b> windows. (Only valid in a tab window.)                                                                                                                                                                             |
| <b>BOTTOM</b>             | Moves the <b>Take Command</b> window to the bottom of the window order. (Only valid in a tab window.)                                                                                                                                                                                                                       |
| <b>DETACH</b>             | Detach <b>TCC</b> from a <b>Take Command</b> tab window.                                                                                                                                                                                                                                                                    |

#### 4.3.182 WMQUERY

**Purpose:** Query the Windows Management Interface

**Format:** WMQUERY [/A /B /C /H] namespace "query string" [index]

**namespace**              The namespace to query

**"query string"**        WQL query string

**index**                  Class instance

[/A\(ll instances\)](#)    [/C\(lasses\)](#)

[/B\(lank\)](#)            [/H\(eader\)](#)

**Usage:**

You can use a single period . for **namespace** to default to **root\cimv2**.

For example, to query the **name** property from the **Win32\_Processor** class:

```
wmiquery root\cimv2 "SELECT name FROM Win32_Processor"
```

To query available classes:

```
wmiquery /A root "select name from __namespace"
```

For more details on what is available, see the WMI and WQL documentation on MSDN ([msdn.microsoft.com](http://msdn.microsoft.com)), and download the "WMI Code Creator" from Microsoft at:

<http://www.microsoft.com/en-us/download/details.aspx?id=8572>

and browse the available namespaces, classes, and properties. WMI Code Creator is also a useful tool for generating WMI queries for use with WMICQUERY.

**Options:**

- /A      Display all class instances starting at "index".
- /B      Separate class instances with a blank line.
- /C      Display all the matching class names for the specified namespace. "**query string**" is the filter to apply to the returned values; it can contain wildcards. For example:

```
wmiquery /c . "win32_q*"
```

- /H      Display a header for class instances.

#### 4.3.183 Y

**Purpose:** Copy standard input to standard output, and then copy the specified file(s) to standard output

**Format:** Y file ...

**file**    The file or list of files to send to standard output.

See also: [TEE](#), [piping](#) and [redirection](#).

**Usage:**

The Y command copies input from standard input (usually the keyboard) to standard output (usually the screen). Once the input ends, the named files are appended to standard output.

For example, to get text from standard input, append the files *MEMO1* and *MEMO2* to it, and send the output to *MEMOS*:

```
y memo1 memo2 > memos
```

The Y command is most useful if you want to add redirected data to the beginning of a file instead of appending it to the end. For example, this command copies the output of DIR, followed by the contents of the file DIREND, to the file DIRALL:

```
dir | y dirend > dirall
```

If you are typing at the keyboard to produce input text for Y, you must enter a **Ctrl-Z** to terminate the input.

When using Y with a pipe you must take into account that the programs on the two ends of the pipe run simultaneously, not sequentially.

See [Piping](#) for more information on pipes.

#### 4.3.184 ZIP

**Purpose:** Add, update, or delete files in a .ZIP archive

**Format:** ZIP [/A:[[-][+]rhsdaecjot] /A /C /CRC /D /En /F /I /Ln /M /O:[-]adegnrstu /P /Q /R /S"password" /T /TEST /U /V /YC /Z"text"] ziparchive [@file] file...

**ziparchive** The zip file to work with  
**file** The files(s) to be added to the zip file

|                                           |                                      |
|-------------------------------------------|--------------------------------------|
| <a href="#">/A:... (attribute switch)</a> | <a href="#">/P(rogress)</a>          |
| <a href="#">/A(dd)</a>                    | <a href="#">/Q(uiet)</a>             |
| <a href="#">/C(ontents)</a>               | <a href="#">/R(ecurse)</a>           |
| <a href="#">/CRC</a>                      | <a href="#">/S"password"</a>         |
| <a href="#">/D(elete)</a>                 | <a href="#">/T (save attributes)</a> |
| <a href="#">/En (encryption type)</a>     | <a href="#">/TEST</a>                |
| <a href="#">/F(reshen)</a>                | <a href="#">/U(pdate)</a>            |
| <a href="#">/I (save descriptions)</a>    | <a href="#">/V(iew)</a>              |
| <a href="#">/Ln (compression level)</a>   | <a href="#">/YC (AES 256)</a>        |
| <a href="#">/M(ove)</a>                   | <a href="#">/Z (comment)</a>         |
| <a href="#">/O:... (sort order)</a>       |                                      |

#### File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

#### Usage:

You can specify a pathname for `ziparchive`. If you don't provide an extension, and the filename as entered doesn't exist, ZIP adds ".zip". If you don't specify an operation, ZIP will default to Add. If you don't specify any arguments, ZIP will display its command dialog.

ZIP will automatically use the Zip64 extensions if the archive is in Zip64 format.

ZIP sets two internal variables:

|                           |                              |
|---------------------------|------------------------------|
| <code>%_zip_files</code>  | The number of files archived |
| <code>%_zip_errors</code> | The number of errors         |

#### Option:

**/=** Display the ZIP command dialog to help you set the filename and command line options.  
 You cannot specify any other arguments on the command line.

**/A:...** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with [@file](#) lists. See [@file lists](#) for details.

You can specify **/A:=** to display a dialog to help you set individual attributes.

- /A** Add the specified file(s) to the zip file. (This is the default.)
- /C** Display (on standard output) the contents of a file in the zip archive.
- /CRC** Display the file CRCs (must be used with **/V**).
- /D** Delete the specified file(s) from the zip file.
- /En** Set the encryption type (0=default, 1=AES 128-bit, 2=AES 192-bit, 3=AES 256-bit).
- /F** Update only those files that currently exist in the zip file, and which are older than the files on disk.
- /I** Save file descriptions (from DESCRIPT.ION or the NTFS description) as the compressed file's "File Comment".
- /Ln** Set the compression level (0 - 6, where 0=no compression, and 6=maximum compression). The default is 4.
- /M** Delete the files from the disk after adding them to the zip file.
- /O:...** Sort the files before processing.

You may use any combination of the sorting options below. If multiple options are used, the files will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

- /P** Display the progress (0 - 100%) for each file as it is zipped.
- /Q** Don't display the files being zipped.
- /R** If the argument is a subdirectory, copy all of the files in that subdirectory and all of its subdirectories to the zip file.
- /S** Use the specified password to encrypt the file(s). If you don't provide a password on the command line, ZIP will prompt you to enter one.
- /T** Save the file attributes (they will be set when the file is extracted).

- /TEST** Test the integrity of the ZIP file (header and contents). Any errors will be displayed on STDERR.
- /U** Update files which either don't exist in the zip, or which are older than the files on disk.
- /V** View the list of files in the zip file (date, time, size, compression ratio, and filename). If the zip file is password protected, ZIP will append a \* after the filename.
- /YC** Use AES 256-bit encryption instead of the default Zip 2.0 encryption.
- /Z"..."** Set the comment for the zip file.

#### 4.3.185 ZIPSFX

**Purpose:** Create a ZIP-compatible self-extracting archive

**Format:** ZIPSFX [/B /C /D"path" /F"file" /Ln /M"message" /R /S"password" /X64] **exarchive** **directory...**

**exarchive** The self-extracting executable

**directory** The directory to be compressed into the self-extracting executable

[/B\(anner\)](#)

[/M\(essage\)](#)

[/C\(aption\)](#)

[/R\(ecurse\)](#)

[/D \(target directory\)](#)

[/S"password"](#)

[/F \(execute after open\)](#)

[/X64 \(64-bit executable\)](#)

[/Ln \(compression level\)](#)

##### **File Selection**

Supports extended [wildcards](#) and [ranges](#),

##### **Usage:**

You can specify a pathname for **exarchive**. If you don't provide an extension, and the filename as entered doesn't exist, ZIPSFX adds ".exe".

ZIPSFX sets two internal variables:

**%\_zipsfx\_files** The number of files archived

**%\_zipsfx\_errors** The number of errors

##### **Option:**

- /B** Banner text to display before beginning the self-extraction.
- /C** Caption for the self-extractor dialogs.
- /D** Target directory for the self-extractor.
- /F** Optional name of the file to execute (open) after the archive is extracted. This must be a relative path to a file in **directory**. If this is set to ".", the folder in which the archive has been decompressed will open in Windows Explorer. If it is set to "" (an empty string), the

extractor will close and take no action.

- /Ln** Set the compression level (0 - 6, where 0=no compression, and 6=maximum compression). The default is 4.
- /M** Message to notify the user that the extraction has completed normally.
- /R** If the argument is a subdirectory, copy all of the files in that subdirectory and all of its subdirectories to the zip file.
- /S** Use the specified password to extract the file(s). If you don't provide a password on the command line, ZIPSFX will prompt you to enter one.
- /X64** Create a 64-bit executable.

#### 4.3.186 7UNZIP

**Purpose:** Extract files from .7Z archives

**Format:** 7UNZIP [/A:[[-][+]rhsdaecjot] /C /CRC /D /E /F /Nt /P /O /Q /S"password" /TEST /U /V] ziparchive path file ...

|                   |                                        |
|-------------------|----------------------------------------|
| <b>ziparchive</b> | The 7Zip file to work with             |
| <b>path</b>       | The path where files will be extracted |
| <b>file</b>       | The file(s) to extract                 |

|                                   |                      |
|-----------------------------------|----------------------|
| <u>/A:.... (attribute switch)</u> | <u>/O(overwrite)</u> |
| <u>/C(ontents)</u>                | <u>/P(ercent)</u>    |
| <u>/CRC</u>                       | <u>/Q(uiet)</u>      |
| <u>/D(irectory)</u>               | <u>/S"password"</u>  |
| <u>/E(xtract)</u>                 | <u>/TEST</u>         |
| <u>/F(reshen)</u>                 | <u>/U(pdate)</u>     |
| <u>/N</u>                         | <u>/V(iew)</u>       |

#### File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

#### Usage:

You can specify a pathname for *ziparchive*. If you don't provide an extension, and the filename as entered doesn't exist, 7UNZIP adds ".7z". If you don't specify an operation, 7UNZIP will default to Extract. If you don't specify any arguments, 7UNZIP will display its command dialog.

7UNZIP supports wildcards for the 7Zip archive name and for the filenames to extract. 7UNZIP will prompt before overwriting existing files. Your options at the prompt are explained in detail under [Page and File Prompts](#).

*path* specifies the path where files will be extracted. If *path* is not specified, files are extracted to the current directory.

7UNZIP sets two internal variables:

%\_7unzip\_files The number of files extracted  
 %\_7unzip\_errors The number of errors

**Option:**

- /= Display the 7UNZIP command dialog to help you set the filename and command line options. You cannot specify any other arguments on the command line.
- /A:... Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with @file lists. See [@file lists](#) for details.  
  
You can specify /A:= to display a dialog to help you set individual attributes.
- /C Display (on standard output) the contents of a file in the 7Zip archive.
- /CRC Display the file CRCs (must be used with /V).
- /D Recreate the directory structure saved in the 7Zip file.
- /E Extract the specified file(s). (This is the default.)
- /F Extract only those files that currently exist in the target folder, and which are older than the file in the 7Zip archive.
- /Nt Don't update the CD / CDD [extended directory search](#) database (JPSTREE.IDX).
- /O Overwrite existing files. 7UNZIP normally prompts before overwriting an existing file; /O will suppress the prompt.
- /P Display the progress (0 - 100%) for each file as it is extracted.
- /Q Don't display filenames as they are extracted.
- /S Use the specified password to extract the file(s) from an encrypted archive. If you don't provide a password on the command line, 7UNZIP will prompt you to enter one.
- /TEST Test the integrity of the 7Zip file (header and contents). Any errors will be displayed on STDERR.
- /U Extract files which either don't exist in the target folder, or which are older than the file in the 7Zip archive.
- /V View the list of files in the archive (date, time, size, and filename). If the 7Zip file is password protected, 7UNZIP will append a \* after the filename.

### 4.3.187 7ZIP

**Purpose:** Add, update, or delete files in a .7Z archive

**Format:** 7ZIP [/A:[[-][+]rhsdaecjot] /A /C /CRC /D /F /Kn /Ln /M /O:[-]adegnrstu /P /Q /R /S"password" /T /TEST /U /V] ziparchive [@file] file...

**ziparchive** The 7zip file to work with  
**file** The files(s) to be added to the 7zip file

|                                            |                                      |
|--------------------------------------------|--------------------------------------|
| <a href="#">/A:.... (attribute switch)</a> | <a href="#">/P(rogress)</a>          |
| <a href="#">/A(dd)</a>                     | <a href="#">/Q(uiet)</a>             |
| <a href="#">/C(ontents)</a>                | <a href="#">/R(ecurse)</a>           |
| <a href="#">/CRC</a>                       | <a href="#">/S"password"</a>         |
| <a href="#">/D(elete)</a>                  | <a href="#">/T (save attributes)</a> |
| <a href="#">/F(reshen)</a>                 | <a href="#">/TEST</a>                |
| <a href="#">/Kn (compression method)</a>   | <a href="#">/U(pdate)</a>            |
| <a href="#">/Ln (compression level)</a>    | <a href="#">/V(iew)</a>              |
| <a href="#">/M(ove)</a>                    |                                      |
| <a href="#">/O:.... (sort order)</a>       |                                      |

### File Selection

Supports [command dialog](#), [attribute switches](#), extended [wildcards](#), [ranges](#), [multiple file names](#), and [include lists](#).

### Usage:

You can specify a pathname for **ziparchive**. If you don't provide an extension, and the filename as entered doesn't exist, 7ZIP adds ".7z". If you don't specify an operation, 7ZIP will default to Add. If you don't specify any arguments, 7ZIP will display its command dialog.

7ZIP sets two internal variables:

|                            |                              |
|----------------------------|------------------------------|
| <code>%_7zip_files</code>  | The number of files archived |
| <code>%_7zip_errors</code> | The number of errors         |

### Option:

- /=** Display the ZIP command dialog to help you set the filename and command line options.  
 You cannot specify any other arguments on the command line.
- /A:....** Select only those files that have the specified attribute(s) set. See [Attribute Switches](#) for information on the attributes which can follow /A:. Do not use /A: with [@file lists](#). See [@file lists](#) for details.  
  
 You can specify **/A:=** to display a dialog to help you set individual attributes.
- /A** Add the specified file(s) to the zip file. (This is the default.)
- /C** Display (on standard output) the contents of a file in the zip archive.
- /CRC** Display the file CRCs (must be used with /V).
- /D** Delete the specified file(s) from the zip file.
- /F** Update only those files that currently exist in the 7zip file, and which are older than the files on disk.

**/Kn** Set the compression method.

- 0 LZMA (default)
- 1 BZip2
- 2 Delta
- 3 Copy (no compression)
- 4 Deflate
- 5 LZMA2

**/Ln** Set the compression level (1 - 5, where 1=minimum compression, and 5=maximum compression). The default is 3.

**/M** Delete the files from the disk after adding them to the 7zip file.

**/O:...** Sort the files before processing.

You may use any combination of the sorting options below. If multiple options are used, the files will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

- n** Sort by filename and extension, unless **e** is explicitly included. This is the default.
- Reverse the sort order for the next sort key
- a** Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension.
- d** Sort by date and time (oldest first); also see **/T:acw**
- e** Sort by extension
- g** Group subdirectories first, then files
- r** Reverse the sort order for all options
- s** Sort by size
- t** Same as **d**
- u** Unsorted

**/P** Display the progress (0 - 100%) for each file as it is zipped.

**/Q** Don't display the files being zipped.

**/R** If the argument is a subdirectory, copy all of the files in that subdirectory and all of its subdirectories to the zip file.

**/S** Use the specified password to encrypt the file(s). If you don't provide a password on the command line, 7ZIP will prompt you to enter one.

**/T** Save the file attributes (they will be set when the file is extracted).

**/TEST** Test the integrity of the 7ZIP file (header and contents). Any errors will be displayed on STDERR.

**/U** Update files which either don't exist in the 7zip, or which are older than the files on disk.

**/V** View the list of files in the 7zip file (date, time, size, and filename). If the 7zip file is password protected, 7ZIP will append a \* after the filename.

## 4.4 Variables & Functions

The environment is a collection of information about your system that every program receives. Each entry in the environment consists of a variable name and a string value.

### Usage

You can automatically substitute the text for the variable name in any command. To create the substitution, include a percent sign % and the variable name on the command line or in an alias or batch file, e.g., **%comspec**. If the name of the variable whose value you want to use is an expression, you can enclose the expression in brackets, e.g., **%[%n]**.

You can create, alter, view, and delete environment variables with the [SET](#), [ESET](#), and [UNSET](#) commands.

A few environment variables have special meanings for **TCC** (they are listed in [System Variables](#) ).

**TCC** also supports two special types of variables:

- ▶ [Internal variables](#) are similar to environment variables, but are interpreted internally by **TCC**, and are not visible in the environment. They provide information about your system for use in batch files and aliases. Some of them provide access to information that may change even during the execution of a single command or batch file.
- ▶ [Variable functions](#) are referenced like environment variables, but perform additional actions like file handling, string manipulation and arithmetic calculations. In addition to the variable functions that are internal to **TCC**, you can use the [FUNCTION](#) command to create your own. These latter ones are referred to as user defined functions or UDFs.

**Note:** **TCC** inherits its initial environment from the process which started it. That process might be Explorer or another existing Windows process which launched the current **TCC** session. Note that if the starting process's environment is changed (through registry modifications, for example) while **TCC** is already running, those changes will not be automatically reflected in **TCC**'s current environment. See the [SET](#) command for details.

You use the [SET](#) command to create a new environment variable. [SET](#) can also modify or delete a single environment variable, or display the value of one or more environment variables. [ESET](#) allows you to edit an environment variable. [UNSET](#) deletes environment variables. For example, you can create a variable named **BACKUP** like this:

```
set BACKUP=*.bak;*.bk
```

If you then type:

```
del %BACKUP%
```

it is equivalent to having type the command:

```
del *.bak;*.bk
```

The environment variable names you use this way may contain any alphabetic or numeric characters, the underscore character \_, and the dollar sign \$. You can force acceptance of other characters by including the full variable name in square brackets, like this: **%[AB##2]**. You can also indirectly reference

environment variables using square brackets. For example `%[%var1]` means "the contents of the variable whose name is stored in **VAR1**".

In addition, **TCC** uses the environment to keep track of the default directory on each drive. Windows only tracks the default directory of the current drive; **TCC** overcomes this limitation by saving the default directory for each drive in the environment, using hidden variable names. Each variable begins with an equal sign followed by the drive letter and a colon (for example, `=C:`). You cannot view or change these variables with the [SET](#) command.

The trailing percent sign that was traditionally required for environment variable names is not usually required by **TCC**, which accept any character that cannot be part of a variable name as the terminator. However, the trailing percent can be used to maintain compatibility with CMD.

The trailing percent sign is needed if you want to append variable values. The following examples show the possible interactions between variables and literal strings. First, create two environment variables called ONE and TWO this way:

```
set ONE=abcd
set TWO=eefgh
```

Now the following combinations produce the output text shown:

| <i>original</i>            | <i>expanded</i>        | <i>method</i>                      |
|----------------------------|------------------------|------------------------------------|
| <code>%ONE%TWO</code>      | <code>abcdTWO</code>   | <code>("%ONE%" + "TWO")</code>     |
| <code>%ONE%TWO%</code>     | <code>abcdTWO</code>   | <code>("%ONE%" + "TWO%")</code>    |
| <code>%ONE%%TWO</code>     | <code>abcdefgh</code>  | <code>("%ONE%" + "%TWO")</code>    |
| <code>%ONE%%TWO%</code>    | <code>abcdefgh</code>  | <code>("%ONE%" + "%TWO%")</code>   |
| <code>%ONE% [TWO]</code>   | <code>abcd[TWO]</code> | <code>("%ONE%" + "[TWO]")</code>   |
| <code>%ONE% [TWO]%</code>  | <code>abcd[TWO]</code> | <code>("%ONE%" + "[TWO]%)</code>   |
| <code>% [ONE] %TWO</code>  | <code>abcdefgh</code>  | <code>("% [ONE]" + "%TWO")</code>  |
| <code>% [ONE] %TWO%</code> | <code>abcdefgh</code>  | <code>("% [ONE]" + "%TWO%")</code> |

If you want to pass a percent sign to a command, or a string which includes a percent sign, you must use two percent signs in a row. Otherwise, the single percent sign will be seen as the beginning of a variable name and will not be passed on to the command. For example, to display the string "We're with you 100%" you would use the command:

```
echo We're with you 100%%
```

You can also use back quotes around the text, rather than a double percent sign. See [Parameter Quoting](#) for details.

Environment variables may contain alias names. **TCC** will substitute the variable value for the name, then check for any alias name which may have been included within the value. For example, the following commands would generate a 2-column directory of the .TXT files:

```
alias d2 dir /2
set cmd=d2
%cmd *.txt
```

For compatibility with some peculiar syntax introduced in recent CMD versions, **TCC** supports:

|                                    |                                                                        |
|------------------------------------|------------------------------------------------------------------------|
| <code>%var:string1:string2%</code> | Substitutes the second string for all instances of the first string in |
|------------------------------------|------------------------------------------------------------------------|

`%var:~x[,y]%`

the variable.  
Returns the substring starting at the xth character position (base 0) and continuing for y characters. If y is not specified, returns the remainder of the string. If x is negative, starts from the end of the string.

For string manipulations, we suggest you rely instead on the much more flexible [Variable Functions](#).

#### 4.4.1 Array Variables

In addition to environment variables, **TCC** also supports up to 4-dimensional array variables. Array variables are defined by the [SETARRAY](#) command, and you assign values to them with the [SET](#) command.

See also [UNSETARRAY](#), [@ARRAYINFO](#), [@EXECARRAY](#) and [@FILEARRAY](#).

#### 4.4.2 System Variables

The variables below have special meaning for **TCC**:

|                                 |                                                                    |
|---------------------------------|--------------------------------------------------------------------|
| <a href="#">CDPATH</a>          | Directory navigation search list                                   |
| <a href="#">CMDLINE</a>         | Command line after full expansion                                  |
| <a href="#">COLORDIR</a>        | Directory colorization specification                               |
| <a href="#">COMSPEC</a>         | Command processor specification                                    |
| <a href="#">FILECOMPLETION</a>  | File completion control variable                                   |
| <a href="#">HISTORYEXCLUDE</a>  | List of commands excluded from the <a href="#">command history</a> |
| <a href="#">PATH</a>            | Executable program location search list                            |
| <a href="#">PATHEXT</a>         | Ordered search list of extensions of executable programs           |
| <a href="#">PROMPT</a>          | Command prompt format specification                                |
| <a href="#">PROMPT2</a>         | Prompt for line continuation                                       |
| <a href="#">RECYCLEEXCLUDE</a>  | List of files excluded from the recycle bin                        |
| <a href="#">TCMD</a>            | <b>Take Command</b> 's pathname                                    |
| <a href="#">TCMDVER</a>         | <b>Take Command</b> 's version number                              |
| <a href="#">TEMP</a>            | Directory for temporary files                                      |
| <a href="#">TITLEPROMPT</a>     | Command processor window title bar specification                   |
| <a href="#">TMP</a>             | Directory for temporary files                                      |
| <a href="#">TREEEXCLUDE</a>     | List of directories excluded from JPSTREE.IDX                      |
| <a href="#">VARIABLEEXCLUDE</a> | Variables to exclude from SET list                                 |

##### 4.4.2.1 CDPATH

**CDPATH** specifies where to search for directories specified in [CD](#), [CDD](#), and [PUSHD](#) commands and in [automatic directory changes](#). See the [CDPATH feature](#) for details.

This feature is maintained for backwards compatibility, but has largely been replaced by [Extended Directory Searches](#).

##### 4.4.2.2 CMDLINE

**CMDLINE** is set by **TCC** to the fully expanded text of the currently executing command line just before invoking any external command (.EXE, .BTM, .BAT or .CMD), unless the command line is prefaced with @ to prevent echoing, in which case **CMDLINE** will be removed.

#### 4.4.2.3 COLORDIR

**COLORDIR** controls directory display colors used by [DIR](#). See [Color-Coded Directories](#) for a complete description of the format of this variable.

#### 4.4.2.4 COMSPEC

Many programs expect the value of **COMSPEC** to contain the full path and name of the current character-mode command processor, e.g.

```
c:\program files\jisoft\tcmd13\tcc.exe
```

**TCC** automatically sets **COMSPEC** to point to **TCC.EXE** on startup. If you need to run a program from **TCC** which utilizes **COMSPEC** to locate a command processor to process commands or batch files that are not compatible with **TCC**, you may set **COMSPEC** to the command processor your program expects before you start it.

#### 4.4.2.5 FILECOMPLETION

**FILECOMPLETION** specifies the files made available during filename completion for selected commands. See [Customizing Filename Completion](#) for a complete description of the format.

#### 4.4.2.6 HISTORYEXCLUDE

**HistoryExclude** specifies which commands should be excluded from the [History List](#). The syntax is:

```
HistoryExclude=cmd1[;cmd2[;cmd3[;...]]]
```

For example, to exclude the [DEL](#) and [FREE](#) commands, the **NotePad** program and the user-defined alias **MYDIR**:

```
HistoryExclude=del;free;notepad;mydir
```

See also: [HISTORY](#).

#### 4.4.2.7 PATH

The **PATH** variable specifies the list of directories that **TCC** will search for executable files that aren't in the current directory. **PATH** is used by some application programs to find their own files. See the [PATH](#) command for a full description of this variable, which can also be changed or modified with [SET](#) and [ESET](#).

**Note:** We strongly recommend that you always leave at least your **WINDOWS** and **SYSTEM32** directories in the **PATH**. The directory where **TCC** resides need not be included in **PATH**.

#### 4.4.2.8 PATHEXT

**PATHEXT** is expected to contain a list of extensions (including a leading period .), separated by semicolons. For example, to replicate the default extension list used by **TCC**:

```
set pathext=.exe;.btm;.bat;.cmd;.vbs;.js;.ws;.rex;.rexx;.pl;.rb;.py;.tcl
```

If you use a command in a batch file or at the command prompt and all of the following are true:

- the [PathExt](#) configuration option is set
- the command is not an alias
- the command is not an internal command

- the command is not a filename with an explicit extension (thus neither an executable extension nor a Windows file association is available)

then **TCC** will search the current directory and then each directory listed in [PATH](#) in turn for a file with its name matching the command and its extension matching one of the extensions in PATHEXT. The current directory is searched first, then the first directory in [PATH](#) is searched first, then the second, looking in each for each of the extensions in PATHEXT in the order listed.

**Caution:** If you set the [PathExt](#) configuration option, and fail to set the PATHEXT variable, path searches without an explicit extension will fail as there will be no extensions for which to search! If you set the PathExt configuration option but do not create or modify the PATHEXT variable, **TCC** will use the one defined by Windows (if any), which will probably not include the .BTM, .REX, .REXX, .PL, .RB, .PY, or .TCL extensions.

#### 4.4.2.9 PROMPT

**PROMPT** defines the command line prompt. It can be set or changed with the [PROMPT](#), [SET](#) and [ESET](#) commands. See the [PROMPT](#) command for details.

See also: [TITLEPROMPT](#).

#### 4.4.2.10 PROMPT2

**PROMPT2** defines the prompt used for line continuations (i.e., when the last character on a line is an escape character, or when there is an open command group). The default is "More? ".

#### 4.4.2.11 RECYCLEEXCLUDE

**RECYCLEEXCLUDE** specifies files to be excluded from the Recycle Bin.

The syntax is:

RecycleExclude=*file1[:file2...]*

*file1, file2, ...* : file specifications, may include wildcards

For example, to exclude \*.lib, \*.obj, and \*.bak files:

RecycleExclude=\*.lib;\*.obj;\*.bak

See also: [DEL / ERASE](#) command and the [Delete to Recycle Bin](#) configuration option.

#### 4.4.2.12 TEMP

**TEMP** specifies the directory where **TCC** should store temporary files, unless the [TMP](#) variable exists. Many other programs also use TEMP to locate where they should place their temporary files.

#### 4.4.2.13 TCMD

**TCMD** is the full pathname of the **Take Command** executable. It is only available to the applications running in **Take Command** tab windows.

#### 4.4.2.14 TCMDVER

**TCMDVER** is the current major version, minor version, and build number of **Take Command**. (For example, "16.00.20".) It is only available to applications started in **Take Command** tab windows.

#### 4.4.2.15 TITLEPROMPT

**TITLEPROMPT** can be used to specify the contents of **TCC**'s window title. Modifying its value changes the displayed title immediately. Unsetting it does NOT affect the title. It may contain the special escape-sequences acceptable in [PROMPT](#), and all internal variables and functions can be used to generate it.

If you have specified a title for a startup tab in **Take Command**, it will override the **TITLEPROMPT** value.

See also: [ACTIVATE](#), [PROMPT](#), [TITLE](#) and [WINDOW](#).

#### 4.4.2.16 TMP

If **TMP** is defined, it specifies the directory where **TCC** should store temporary files (overriding [TEMP](#)). Some other programs also use **TMP** to define where they should place their temporary files.

#### 4.4.2.17 TREEEXCLUDE

**TreeExclude** specifies which drives and directories to ignore when updating the **JPSTREE.IDX** file. The syntax is:

```
TreeExclude=dir1[;dir2[;dir3[;...]]]
```

Any specified drive/directory and all of its subdirectories will be excluded from **JPSTREE.IDX** update. For example, to exclude everything in **c:\windows**, **d:\temp\temp2**, and everything on drive **g**:

```
TreeExclude=c:\windows;d:\temp\temp2;g:\
```

Setting **TreeExclude** to the base directory of the target of a directory tree copy can speed up the copying considerably.

See also: [Extended Directory Searches](#) and [CDD](#).

#### 4.4.2.18 VARIABLEEXCLUDE

**VariableExclude** specifies which environment variables should be excluded from the SET list. The syntax is:

```
VariableExclude=var1[;var2[;var3[;...]]]
```

For example, to exclude the SESSIONNAME, TMP, USERDOMAIN, and USERNAME variables:

```
SET VariableExclude=sessionname;tmp;userdomain;username
```

See also: [SET](#).

### 4.4.3 CMD.EXE Variables

CMD has some built-in variables (i.e., which are treated as environment variables but which do not exist in the environment):

**CD** - the current directory (see also [CWD](#)).

**CMDCMDLINE** - the command line that started the command processor.

**CMDEXTVERSION** - the command extensions internal version number.

**DATE** - the current system date (see also [DATE](#)).

**RANDOM** - a random number between 0 and 32767 (see also [@RANDOM](#)).

**TIME** - the current system time (see also [TIME](#)).

**TCC** supports all of these built-in variables. (In **TCC**, **CMDEXTVERSION** will always return 2.)

The variables below are used by some Microsoft command processors, but are ignored by **TCC**. To see their usage by Microsoft and the alternate methods to achieve the same purpose in **TCC**, review:

|                |                                                      |
|----------------|------------------------------------------------------|
| <b>COPYCMD</b> | CMD default options for <a href="#">COPY</a> command |
| <b>DIRCMD</b>  | CMD default options for <a href="#">DIR</a> command  |

#### 4.4.3.1 **COPYCMD**

The **COPYCMD** variable is used by some versions of CMD to hold default options for the [COPY](#) command. **TCC** does not directly support this variable, i.e., its value has no affect on internal commands. In general, it is more efficient to define several aliases, each including a different combination of options. For example, if you want the COPY command to default to prompting you before overwriting an existing file, you could use this alias:

```
alias COPY='*copy /r'
```

If you wish to use or create a **COPYCMD** variable for compatibility with CMD, you can define an alias to append the contents of that variable to the COPY command:

```
alias COPY='*copy %copycmd'
```

Now each time the COPY alias is executed, the current value of **COPYCMD** will modify the execution of the COPY command.

#### 4.4.3.2 **DIRCMD**

The **DIRCMD** variable is used by some versions of CMD to hold default options for the [DIR](#) command. **TCC** does not directly support this variable, i.e., its value has no affect on internal commands. In general, it is more efficient to define several aliases, each including a different combination of options. For example, if you want the DIR command to default to a 2-column display with a vertical sort and a pause at the end of each page, you could use this alias:

```
alias DIR='*dir /2 /p /v`I
```

If you wish to use or create a **DIRCMD** variable for compatibility with CMD, you can define the alias to append the contents of that variable to the DIR command:

```
alias DIR='*dir %dircmd'
```

Now each time the DIR alias is executed, the current value of **DIRCMD** will modify the execution of the DIR command.

#### 4.4.3.3 **String substitution**

For compatibility with some peculiar syntax introduced in recent CMD versions, **TCC** supports:

|                       |                                                                                                                                                                                                                         |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| %var:string1=string2% | Substitutes the second string for all instances of the first string in the variable.                                                                                                                                    |
| %var:~x[,y]%          | Returns the substring starting at the xth character position (base 0) and continuing for y characters. If y is not specified, returns the remainder of the string. If x is negative, starts from the end of the string. |

For string manipulations, we suggest you rely instead on the much more flexible [Variable Functions](#).

#### 4.4.4 Variables

**Internal variables** are special variables built into **TCC** to provide information about your system. They are not stored in the environment, but can be accessed as if they were environment variables in interactive commands, aliases, and batch files.

The values of these variables are stored internally in **TCC**, and cannot be changed with the [SET](#), [UNSET](#), [ESET](#) or any other command. The DEFINED status test will always fail, too. You can override any of these variables by defining a new environment variable with the same name. The internal variable can be made available again by unsetting the identically name environment variable. The names of ALL internal variables (except the pseudovariables errorlevel, ?, ??, +, and =) begin with an underscore character to make it easier to distinguish them and to avoid accidentally overriding them.

These internal variables are often used in batch files and aliases to examine system resources and adjust to the current computer settings. You can examine the contents of any internal variable (except % = and %+) from the command line with a command like this:

```
echo %variablename
```

Variables which return a file or directory name from a volume that supports long filenames return it in the same case as it is stored. Returned names are not quoted automatically, you must add the quotes yourself if they are required by the syntax in which you use them.

Some variables return values based on information provided by your operating system. These variables will only return correct information if the operating system provides it. For example, [\\_BATTERY](#) will not return accurate results if your operating system and Advanced Power Management drivers do not provide correct information on battery status to **TCC**.

For a list of internal variables organized by general categories of use, see [Internal Variables by Category](#).

#### Examples

You can use internal variables in a wide variety of ways depending on your needs. Here are just a couple of examples:

Store the current date and time in a file, then save the output of a DIR command in the same file:

```
echo Directory as of %_date %_time > dirsave
dir >> dirsave
```

Use the [IFF](#) command to check whether there are enough resources free before running an application:

```
iff %_GDIFREE lt 40 then
 echo Not enough GDI resources!
```

```

 quit
else
 d:\mydir\myapp
endiff

```

Call another batch file if today is Monday:

```
if "%_DOW" == "Mon" call c:\cleanup\weekly.bat
```

#### 4.4.4.1 Variables by Name

|                      |                                              |
|----------------------|----------------------------------------------|
| <a href="#">+</a>    | Substitutes the <b>TCC</b> command separator |
| <a href="#">=</a>    | Substitutes the <b>TCC</b> escape character  |
| <a href="#">!</a>    | Last argument of the previous command        |
| <a href="#">?</a>    | Exit code, last external program             |
| <a href="#">4ver</a> | <b>TCC</b> version                           |
| <a href="#">?</a>    | Exit code, last internal command             |

|                          |                               |
|--------------------------|-------------------------------|
| <a href="#">acstatus</a> | AC line status                |
| <a href="#">admin</a>    | 1 if an administrator, else 0 |
| <a href="#">afswcell</a> | OpenAFS workstation cell      |
| <a href="#">alt</a>      | Alt key depressed: 1, else 0  |
| <a href="#">ansi</a>     | ANSI X3.64 status             |

|                                |                                               |
|--------------------------------|-----------------------------------------------|
| <a href="#">batch</a>          | Batch nesting level                           |
| <a href="#">batchline</a>      | Line number in current batch file.            |
| <a href="#">batchname</a>      | Full path and filename of current batch file. |
| <a href="#">batchtype</a>      | Type of the current batch file                |
| <a href="#">battery</a>        | Battery status                                |
| <a href="#">batterylife</a>    | Remaining battery life, seconds               |
| <a href="#">batterypercent</a> | Remaining battery life, %                     |
| <a href="#">bdebugger</a>      | Batch debugger active: 1, else 0              |
| <a href="#">bg</a>             | Background color at cursor position           |
| <a href="#">boot</a>           | Boot drive letter, without a colon            |
| <a href="#">build</a>          | Build number                                  |

|                             |                                              |
|-----------------------------|----------------------------------------------|
| <a href="#">capslock</a>    | CapsLock on: 1, else 0                       |
| <a href="#">cdroms</a>      | List of the CD-ROM drives                    |
| <a href="#">childpid</a>    | Process ID of most recent child process      |
| <a href="#">ci</a>          | Current text cursor shape in insert mode     |
| <a href="#">cmdline</a>     | Current command line                         |
| <a href="#">cmdproc</a>     | Command processor name                       |
| <a href="#">cmdspec</a>     | Full pathname of command processor           |
| <a href="#">co</a>          | Current text cursor shape in overstrike mode |
| <a href="#">codepage</a>    | Current code page number                     |
| <a href="#">column</a>      | Current cursor column                        |
| <a href="#">columns</a>     | Virtual screen width                         |
| <a href="#">consoleb</a>    | Handle to console screen buffer              |
| <a href="#">consolepids</a> | Process IDs attached to this console         |
| <a href="#">country</a>     | Current country code                         |
| <a href="#">cpu</a>         | CPU type                                     |

|                          |                                             |
|--------------------------|---------------------------------------------|
| <a href="#">cpuusage</a> | CPU time usage (percent)                    |
| <a href="#">ctrl</a>     | Ctrl key depressed: 1, else 0               |
| <a href="#">cwd</a>      | Current drive and directory                 |
| <a href="#">cwds</a>     | Current drive and directory with trailing \ |
| <a href="#">cwp</a>      | Current directory                           |
| <a href="#">cwps</a>     | Current directory with trailing \           |

|                           |                                            |
|---------------------------|--------------------------------------------|
| <a href="#">date</a>      | Current date                               |
| <a href="#">datetime</a>  | Current date and time, yyyyMMddhhmmss      |
| <a href="#">day</a>       | Current day of the month                   |
| <a href="#">detachpid</a> | Process ID of most recent detached process |
| <a href="#">disk</a>      | Current drive                              |
| <a href="#">dname</a>     | Name of the description file.              |
| <a href="#">dos</a>       | Operating system type                      |
| <a href="#">dosver</a>    | Operating system version                   |
| <a href="#">dow</a>       | Current day of the week, English, short    |
| <a href="#">dowf</a>      | Current day of the week, English, full     |
| <a href="#">dowi</a>      | Current day of the week as an integer      |
| <a href="#">doy</a>       | Current day of the year                    |
| <a href="#">drives</a>    | List of the existing drives                |
| <a href="#">dst</a>       | Daylight savings time: 1, else 0           |
| <a href="#">dvds</a>      | List of the DVD drives                     |

|                            |                                                               |
|----------------------------|---------------------------------------------------------------|
| <a href="#">echo</a>       | Echo turned on: 1, else 0                                     |
| <a href="#">editmode</a>   | 0 if in overstrike mode, 1 if in insert mode                  |
| <a href="#">elevated</a>   | 1 if the <b>TCC</b> process is elevated (Vista or later only) |
| <a href="#">errorlevel</a> | Exit code, last external program                              |
| <a href="#">execarray</a>  | Array elements assigned by the last @EXECSTR                  |
| <a href="#">execstr</a>    | @EXECSTR return code                                          |
| <a href="#">exit</a>       | <b>TCC</b> exit return code                                   |
| <a href="#">expansion</a>  | SETDOS /X value                                               |

|                           |                                     |
|---------------------------|-------------------------------------|
| <a href="#">fg</a>        | Foreground color at cursor position |
| <a href="#">ft perror</a> | Last FTP error code                 |

|                           |                                     |
|---------------------------|-------------------------------------|
| <a href="#">hdrvives</a>  | List of the fixed drives            |
| <a href="#">hlogfile</a>  | Current history log file name       |
| <a href="#">host</a>      | Host name of local computer         |
| <a href="#">hour</a>      | Current hour                        |
| <a href="#">hwprofile</a> | Windows hardware profile if defined |

|                           |                                                     |
|---------------------------|-----------------------------------------------------|
| <a href="#">ide</a>       | In the IDE / debugger: 1, else 0                    |
| <a href="#">idleticks</a> | Milliseconds since the last user input              |
| <a href="#">idow</a>      | Current day of the week, local language, short      |
| <a href="#">idowf</a>     | Current day of the week, local language, full       |
| <a href="#">iftp</a>      | IFTP session active: 1, else 0                      |
| <a href="#">iftps</a>     | IFTPS session active: 1, else 0                     |
| <a href="#">imonth</a>    | Current month name, local language, short           |
| <a href="#">imonthf</a>   | Current month name, local language, full            |
| <a href="#">ininame</a>   | Full pathname of the current INI file               |
| <a href="#">insert</a>    | Current input editor state (0=overstrike, 1=insert) |

|                               |                                                           |
|-------------------------------|-----------------------------------------------------------|
| <a href="#">ip</a>            | IP address(es) of local computer.                         |
| <a href="#">ipadapter</a>     | Index of the current adapter                              |
| <a href="#">ipadapters</a>    | Number of adapters in the system                          |
| <a href="#">iparproxy</a>     | Returns 1 if the local computer is acting as an ARP proxy |
| <a href="#">ipdns</a>         | Returns 1 if DNS is enabled on the local computer         |
| <a href="#">ipdnsserver</a>   | The default DNS server for the local computer             |
| <a href="#">iprouting</a>     | Returns 1 if routing is enabled on the local computer     |
| <a href="#">isftp</a>         | IFTTP SSH session active: 1, else 0                       |
| <a href="#">isodate</a>       | Current date in ISO 8601 format                           |
| <a href="#">isodowi</a>       | ISO 8601 numeric day of week                              |
| <a href="#">isowdate</a>      | ISO 8601 current week date (yyyy-Www-d)                   |
| <a href="#">isoweek</a>       | ISO 8601 week of year                                     |
| <a href="#">isowyear</a>      | ISO 8601 week date year                                   |
| <br><a href="#">kbhit</a>     | Keyboard input character is waiting: 1, else 0            |
| <br><a href="#">lalt</a>      | left Alt key depressed: 1, else 0                         |
| <a href="#">lastdir</a>       | Previous directory (from directory history)               |
| <a href="#">lastdisk</a>      | Last valid drive                                          |
| <a href="#">lctrl</a>         | Left Ctrl key depressed: 1, else 0                        |
| <a href="#">logfile</a>       | Current log file name                                     |
| <a href="#">lshift</a>        | Left Shift key depressed: 1, else 0                       |
| <br><a href="#">minute</a>    | Current minute                                            |
| <a href="#">monitors</a>      | Number of monitors                                        |
| <a href="#">month</a>         | Current month of the year as integer                      |
| <a href="#">monthf</a>        | Current month of the year, English, full                  |
| <br><a href="#">numlock</a>   | NumLock on: 1, else 0                                     |
| <br><a href="#">openafs</a>   | OpenAFS installed: 1, otherwise 0                         |
| <a href="#">osbuild</a>       | Windows build number                                      |
| <br><a href="#">parent</a>    | Name of the parent process                                |
| <a href="#">pid</a>           | <b>TCC</b> process ID (numeric)                           |
| <a href="#">pipe</a>          | Current process is running in a pipe: 1, else 0           |
| <a href="#">ppid</a>          | Process ID of parent process                              |
| <br><a href="#">ralt</a>      | Right Alt key depressed: 1, else 0                        |
| <a href="#">rctrl</a>         | Right Ctrl key depressed: 1, else 0                       |
| <a href="#">ready</a>         | List of accessible drives                                 |
| <a href="#">registered</a>    | Registered user name                                      |
| <a href="#">row</a>           | Current cursor row                                        |
| <a href="#">rows</a>          | Screen height                                             |
| <a href="#">rshift</a>        | Right Shift key depressed: 1, else 0                      |
| <br><a href="#">scrollock</a> | ScrollLock on: 1, else 0                                  |
| <a href="#">second</a>        | Current second                                            |
| <a href="#">selected</a>      | Selected text in the <b>TCC</b> tab window                |
| <a href="#">serialports</a>   | Display available serial ports (COM1 - COMn)              |
| <a href="#">service</a>       | TCC is a service: 1, else 0                               |

|                  |                                                |
|------------------|------------------------------------------------|
| <u>shell</u>     | Shell level                                    |
| <u>shift</u>     | Shift key depressed: 1, else 0                 |
| <u>shortcut</u>  | Pathname of shortcut that started this process |
| <u>shralias</u>  | SHRALIAS is loaded: 1, else 0                  |
| <u>startpath</u> | Startup directory of current shell.            |
| <u>startpid</u>  | Process ID of most recent STARTed process      |
| <u>stdin</u>     | STDIN redirected: 0, else 1                    |
| <u>stdout</u>    | STDOUT redirected: 0, else 1                   |
| <u>stderr</u>    | STDERR redirected: 0, else 1                   |
| <u>stzn</u>      | Name of time zone for standard time            |
| <u>stzo</u>      | Offset in minutes from UTC for standard time   |
| <u>syserr</u>    | Latest Windows error code                      |

|                    |                                                                |
|--------------------|----------------------------------------------------------------|
| <u>tctabactive</u> | <b>TCC</b> is active <b>Take Command</b> tab window: 1; else 0 |
| <u>tcexit</u>      | Pathname to TCEXIT.*                                           |
| <u>tcfilter</u>    | Current filter in the List view window                         |
| <u>tcfolder</u>    | Selected folder in the Folders window                          |
| <u>tclistview</u>  | Selected entries in the List View window                       |
| <u>tcstart</u>     | Pathname of the active TCSTART.*                               |
| <u>tctab</u>       | Running inside Take Command: 1; else 0                         |
| <u>tctabs</u>      | Current number of <b>Take Command</b> tab windows              |
| <u>time</u>        | Current time                                                   |
| <u>transient</u>   | Current process is a transient shell: 1, else 0                |
| <u>tzn</u>         | Name of current time zone                                      |
| <u>tzo</u>         | Offset in minutes from UTC for current time zone               |

|                    |                                                     |
|--------------------|-----------------------------------------------------|
| <u>unicode</u>     | Shell uses unicode for redirected output: 1, else 0 |
| <u>utctime</u>     | Current UTC time                                    |
| <u>utcdate</u>     | Current UTC date                                    |
| <u>utcdatetime</u> | Current UTC date and time                           |
| <u>utchour</u>     | Current UTC hour                                    |
| <u>utcisodate</u>  | Current UTC date in ISO format                      |
| <u>utcminute</u>   | Current UTC minute                                  |
| <u>utcsecond</u>   | Current UTC second                                  |

|                   |                                                               |
|-------------------|---------------------------------------------------------------|
| <u>vermajor</u>   | <b>TCC</b> major version                                      |
| <u>verminor</u>   | <b>TCC</b> minor version                                      |
| <u>version</u>    | <b>TCC</b> version in major.minor format (i.e., <b>16.0</b> ) |
| <u>virtualbox</u> | Running inside VirtualBox: 1; else 0                          |
| <u>virtualpc</u>  | Running inside VirtualPC: 1; else 0                           |
| <u>vmware</u>     | Running inside VMWare: 1; else 0                              |
| <u>vpxpixels</u>  | Virtual screen horizontal size                                |
| <u>wypixels</u>   | Virtual screen vertical size                                  |

|                    |                                        |
|--------------------|----------------------------------------|
| <u>windir</u>      | Windows directory pathname             |
| <u>winfgwindow</u> | Title of foreground window.            |
| <u>winname</u>     | Name of local computer                 |
| <u>winsysdir</u>   | Windows system directory pathname      |
| <u>winticks</u>    | Milliseconds since Windows was started |
| <u>wintitle</u>    | Current window title                   |
| <u>winuser</u>     | Name of current user.                  |

|                          |                                 |
|--------------------------|---------------------------------|
| <a href="#">winver</a>   | Windows version number          |
| <a href="#">wow64</a>    | Running inside WOW64: 1; else 0 |
| <a href="#">wow64dir</a> | System WOW64 directory          |

|                         |                                                             |
|-------------------------|-------------------------------------------------------------|
| <a href="#">x64</a>     | 1 if <b>TCC</b> is the x64 (64-bit) version                 |
| <a href="#">xmouse</a>  | Column of last mouse click                                  |
| <a href="#">xpixels</a> | Physical screen horizontal size in pixels                   |
| <a href="#">xwindow</a> | Width of <b>Take Command</b> or <b>TCC</b> window in pixels |

|                         |                                                              |
|-------------------------|--------------------------------------------------------------|
| <a href="#">year</a>    | Current year                                                 |
| <a href="#">ymouse</a>  | Row of last mouse click                                      |
| <a href="#">ypixels</a> | Physical screen vertical size in pixels                      |
| <a href="#">ywindow</a> | Height of <b>Take Command</b> or <b>TCC</b> window in pixels |

#### 4.4.4.2 Variables by Category

- ▶ [TCC status](#)
- ▶ [Compatibility](#)
- ▶ [Dates and times](#)
- ▶ [Drives and directories](#)
- ▶ [Error codes](#)
- ▶ [Hardware status](#)
- ▶ [Operating system and software status](#)
- ▶ [Screen, color, and cursor](#)

The list below gives a one-line description of all [Internal Variables](#) and a cross reference which selects a separate help topic on that variable. Many variables are simple enough that the one-line description is probably sufficient, but in most cases you should check for any additional information in the cross referenced explanation if you are not already familiar with a variable. You can also obtain help on any function with a **HELP variableName** command at the prompt. See the [HELP](#) command for details

#### Hardware status

|                                |                                                       |
|--------------------------------|-------------------------------------------------------|
| <a href="#">acstatus</a>       | AC line status                                        |
| <a href="#">alt</a>            | Alt key depressed                                     |
| <a href="#">battery</a>        | Battery status                                        |
| <a href="#">batterylife</a>    | Remaining battery life, seconds                       |
| <a href="#">batterypercent</a> | Remaining battery life, %                             |
| <a href="#">capslock</a>       | CapsLock on: 1, otherwise 0                           |
| <a href="#">cpu</a>            | CPU type                                              |
| <a href="#">cpuusage</a>       | CPU time usage (percent)                              |
| <a href="#">ctrl</a>           | Ctrl key depressed: 1, otherwise 0                    |
| <a href="#">kbhit</a>          | A keyboard input character is waiting: 1, otherwise 0 |
| <a href="#">lalt</a>           | left Alt key depressed: 1, otherwise 0                |
| <a href="#">lctrl</a>          | left Ctrl key depressed: 1, otherwise 0               |
| <a href="#">lshift</a>         | left Shift key depressed: 1, otherwise 0              |
| <a href="#">numlock</a>        | NumLock on:r 1, otherwise 0                           |
| <a href="#">ralt</a>           | right Alt key depressed: 1, otherwise 0               |
| <a href="#">rctrl</a>          | right Ctrl key depressed: 1, otherwise 0              |
| <a href="#">rshift</a>         | right Shift key depressed: 1, otherwise 0             |
| <a href="#">scrolllock</a>     | ScrollLock on: 1, otherwise 0                         |

|                    |                                     |
|--------------------|-------------------------------------|
| <code>shift</code> | Shift key depressed: 1, otherwise 0 |
|--------------------|-------------------------------------|

**Operating system and software status**

|                          |                                                               |
|--------------------------|---------------------------------------------------------------|
| <code>!</code>           | Last argument of previous command                             |
| <code>admin</code>       | 1 if administrator; else 0                                    |
| <code>ansi</code>        | ANSI X3.64 status                                             |
| <code>boot</code>        | Boot drive letter, without a colon                            |
| <code>codepage</code>    | Current code page number                                      |
| <code>country</code>     | Current country code                                          |
| <code>dos</code>         | Operating system type                                         |
| <code>dosver</code>      | Operating system version                                      |
| <code>elevated</code>    | 1 if the <b>TCC</b> process is elevated (Vista or later only) |
| <code>host</code>        | Host name of local computer.                                  |
| <code>hwprofile</code>   | Windows hardware profile if defined                           |
| <code>idleticks</code>   | Milliseconds since last user input                            |
| <code>ip</code>          | IP address(es) of local computer.                             |
| <code>ipadapter</code>   | Index of the current adapter                                  |
| <code>ipadapters</code>  | Number of adapters in the system                              |
| <code>iparpproxy</code>  | Returns 1 if the local computer is acting as an ARP proxy     |
| <code>ipdns</code>       | Returns 1 if DNS is enabled on the local computer             |
| <code>ipdnsserver</code> | The default DNS server for the local computer                 |
| <code>iprouting</code>   | Returns 1 if routing is enabled on the local computer         |
| <code>osbuild</code>     | Windows build number                                          |
| <code>serialports</code> | Display available serial ports (COM1 - COMn)                  |
| <code>tctab</code>       | Running inside Take Command: 1; else 0                        |
| <code>virtualbox</code>  | Running inside VirtualBox: 1; else 0                          |
| <code>virtualpc</code>   | Running inside VirtualPC: 1; else 0                           |
| <code>vmware</code>      | Running inside VMWare: 1; else 0                              |
| <code>windir</code>      | Windows directory pathname                                    |
| <code>winfgwindow</code> | Title of foreground window.                                   |
| <code>winname</code>     | Name of local computer                                        |
| <code>winsysdir</code>   | Windows system directory pathname                             |
| <code>winticks</code>    | Milliseconds since Windows was started                        |
| <code>wintitle</code>    | Current window title                                          |
| <code>winuser</code>     | Name of current user.                                         |
| <code>winver</code>      | Windows version number                                        |
| <code>wow64</code>       | Running in Windows x64: 1; else 0                             |
| <code>wow64dir</code>    | System WOW64 directory                                        |
| <code>x64</code>         | 1 if <b>TCC</b> is the x64 (64-bit) version                   |

**TCC status**

|                        |                                              |
|------------------------|----------------------------------------------|
| <code>4ver</code>      | <b>TCC</b> version                           |
| <code>batch</code>     | Batch nesting level                          |
| <code>batchline</code> | Line number in current batch file.           |
| <code>batchname</code> | Full path and filename of current batch file |
| <code>batchtype</code> | Type of the current batch file               |
| <code>bdebugger</code> | Batch debugger active: 1, otherwise 0        |
| <code>build</code>     | Build number                                 |
| <code>childpid</code>  | Process ID of most recent child process      |
| <code>cmdline</code>   | Current command line                         |

|             |                                                               |
|-------------|---------------------------------------------------------------|
| cmdproc     | Command processor name                                        |
| cmdsproc    | Full pathname of command processor                            |
| consoleb    | Handle to the active console screen buffer                    |
| detachpid   | Process ID of most recent detached process                    |
| dname       | Name of the description file.                                 |
| echo        | Echo status                                                   |
| editmode    | Default insert mode: 1; else 0                                |
| execarray   | Array elements assigned by the last @EXECARRAY                |
| exit        | The <b>TCC</b> exit code                                      |
| expansion   | Current expansion mode (SETDOS /X)                            |
| hlogfile    | Current history log file name                                 |
| ide         | In the IDE / debugger: 1, else 0                              |
| iftp        | IFTP session active: 1, otherwise 0                           |
| iftps       | IFTPS session active: 1, otherwise 0                          |
| isftp       | ISFTP (SSH) session active: 1, otherwise 0                    |
| ininame     | Full pathname of the current INI file                         |
| insert      | Current input editor state (0=overstrike, 1=insert)           |
| logfile     | Current log file name                                         |
| parent      | Name of the parent process                                    |
| pid         | The <b>TCC</b> process ID (numeric)                           |
| pipe        | Current process is running in a pipe: 1, otherwise 0          |
| ppid        | Process ID of parent process                                  |
| registered  | Registered user name                                          |
| selected    | Selected text in current tab window                           |
| service     | <b>TCC</b> is a service: 1, else 0                            |
| shell       | Shell level                                                   |
| shortcut    | Pathname of shortcut that started this process                |
| shralias    | SHRALIAS is loaded: 1, otherwise 0                            |
| startpath   | Startup directory of current shell.                           |
| startpid    | Process ID of most recent STARTed process                     |
| stdin       | STDIN redirected: 0, otherwise 1                              |
| stdout      | STDOUT redirected: 0, otherwise 1                             |
| stderr      | STDERR redirected: 0, otherwise 1                             |
| tctabactive | <b>TCC</b> is the active <b>Take Command</b> tab: 1, else 0   |
| tcexit      | Current value of TCEXIT.*                                     |
| tcfilter    | Current filter in List view window                            |
| tcfolder    | Selected folder in Folders window                             |
| tclistview  | Selected entries in the List View window                      |
| tcstart     | Current value of TCSTART.*                                    |
| tctabs      | Current number of <b>Take Command</b> tab windows             |
| transient   | Current process is a transient shell: 1, otherwise 0          |
| unicode     | <b>TCC</b> uses unicode for redirected output: 1, otherwise 0 |
| vermajor    | <b>TCC</b> major version                                      |
| verminor    | <b>TCC</b> minor version                                      |
| version     | <b>TCC</b> version in major.minor format (i.e., <b>16.0</b> ) |

**Screen, color, and cursor**

|         |                                              |
|---------|----------------------------------------------|
| bg      | Background color at cursor position          |
| ci      | Current text cursor shape in insert mode     |
| co      | Current text cursor shape in overstrike mode |
| column  | Current cursor column                        |
| columns | Virtual screen width                         |
| fg      | Foreground color at cursor position          |

|          |                                                              |
|----------|--------------------------------------------------------------|
| monitors | Number of monitors                                           |
| row      | Current cursor row                                           |
| rows     | Screen height                                                |
| selected | Selected text in current tab window                          |
| vpxixels | Virtual screen horizontal size                               |
| wpixels  | Virtual screen vertical size                                 |
| xmouse   | Column of last mouse click                                   |
| xpixels  | Physical screen horizontal size in pixels                    |
| xwindow  | Width of <b>Take Command</b> or <b>TCC</b> window in pixels  |
| ymouse   | Row of last mouse click                                      |
| ypixels  | Physical screen vertical size in pixels                      |
| ywindow  | Height of <b>Take Command</b> or <b>TCC</b> window in pixels |

**Drives and directories**

|          |                                             |
|----------|---------------------------------------------|
| afswcell | OpenAFS workstation cell                    |
| cdroms   | List of CD-ROM drives                       |
| cwd      | Current drive and directory                 |
| cwd\$    | Current drive and directory with trailing \ |
| cwp      | Current directory                           |
| cwp\$    | Current directory with trailing \           |
| disk     | Current drive                               |
| drives   | List of all available drives                |
| dvds     | List of DVD drives                          |
| hdrives  | List of hard (fixed) drives                 |
| lastdir  | Previous directory (from directory history) |
| lastdisk | Last valid drive                            |
| openafs  | OpenAFS service installed: 1, otherwise 0   |
| ready    | List of ready (accessible) drives           |

**Dates and times**

|          |                                                |
|----------|------------------------------------------------|
| date     | Current date                                   |
| datetime | Current date and time, yyyyMMddhhmmss          |
| day      | Current day of the month                       |
| dow      | Current day of the week, English, short        |
| dowf     | Current day of the week, English, full         |
| dowi     | Current day of the week as an integer          |
| doy      | Current day of the year                        |
| dst      | Daylight savings time: 1, else 0               |
| hour     | Current hour                                   |
| idow     | Current day of the week, local language, short |
| idowf    | Current day of the week, local language, full  |
| imonth   | Current month name, local language, short      |
| imonthf  | Current month name, local language, full       |
| isodate  | Current date in ISO 8601 format                |
| isodowi  | ISO 8601 numeric day of week                   |
| isowdate | ISO 8601 current week date (yyyy-Www-d)        |
| isoweek  | ISO 8601 week of year                          |
| isowyear | ISO 8601 week date year                        |
| minute   | Current minute                                 |
| month    | Current month of the year as integer           |
| monthf   | Current month of the year, English, full       |
| second   | Current second                                 |

|                             |                                                  |
|-----------------------------|--------------------------------------------------|
| <a href="#">stzn</a>        | Name of time zone for standard time              |
| <a href="#">stzo</a>        | Offset in minutes from UTC for standard time     |
| <a href="#">time</a>        | Current time                                     |
| <a href="#">tzn</a>         | Name of current time zone                        |
| <a href="#">tzo</a>         | Offset in minutes from UTC for current time zone |
| <a href="#">utctime</a>     | Current UTC time                                 |
| <a href="#">utcdate</a>     | Current UTC date                                 |
| <a href="#">utcdatetime</a> | Current UTC date and time                        |
| <a href="#">utchour</a>     | Current UTC hour                                 |
| <a href="#">utcisodate</a>  | Current UTC date in ISO format                   |
| <a href="#">utcminute</a>   | Current UTC minute                               |
| <a href="#">utcsecond</a>   | Current UTC second                               |
| <a href="#">year</a>        | Current year                                     |

**Error codes**

|                            |                                  |
|----------------------------|----------------------------------|
| <a href="#">?</a>          | Exit code, last external program |
| <a href="#">?</a>          | Exit code, last internal command |
| <a href="#">errorlevel</a> | Exit code, last external program |
| <a href="#">execstr</a>    | Last @EXECSTR return code        |
| <a href="#">ftperror</a>   | Last FTP error code              |
| <a href="#">syserr</a>     | Latest Windows error code        |

**Compatibility**

|                   |                                             |
|-------------------|---------------------------------------------|
| <a href="#">=</a> | Substitutes the <b>TCC</b> escape character |
| <a href="#">+</a> | Substitutes command separator               |

**4.4.4.3 Command Variables**

Most of the **TCC** file handling commands set internal variables with their results (for example, the number of files processed and the number of errors). See the help for the specific command for more details on the variables.

[7unzip\\_files](#)  
[7unzip\\_errors](#)  
[7zip\\_files](#)  
[7zip\\_errors](#)  
[attrib\\_dirs](#)  
[attrib\\_errors](#)  
[attrib\\_files](#)  
[copy\\_dirs](#)  
[copy\\_errors](#)  
[copy\\_files](#)  
[del\\_dirs](#)  
[del\\_errors](#)  
[del\\_files](#)  
[dir\\_dirs](#)  
[dir\\_errors](#)  
[dir\\_files](#)  
[do\\_dirs](#)  
[do\\_errors](#)  
[do\\_files](#)  
[do\\_loop](#)

[ffind\\_errors](#)  
[ffind\\_files](#)  
[ffind\\_matches](#)  
[for\\_errors](#)  
[for\\_files](#)  
[head\\_errors](#)  
[head\\_files](#)  
[md\\_dirs](#)  
[md\\_errors](#)  
[mklink\\_errors](#)  
[mklink\\_links](#)  
[mklnk\\_errors](#)  
[mklnk\\_links](#)  
[move\\_dirs](#)  
[move\\_errors](#)  
[move\\_files](#)  
[pdir\\_dirs](#)  
[pdir\\_errors](#)  
[pdir\\_files](#)  
[rd\\_dirs](#)  
[rd\\_errors](#)  
[ren\\_dirs](#)  
[ren\\_errors](#)  
[ren\\_files](#)  
[sync\\_dirs](#)  
[sync\\_errors](#)  
[sync\\_files](#)  
[tail\\_errors](#)  
[tail\\_files](#)  
[tar\\_errors](#)  
[tar\\_files](#)  
[touch\\_dirs](#)  
[touch\\_errors](#)  
[touch\\_files](#)  
[type\\_errors](#)  
[type\\_files](#)  
[untar\\_errors](#)  
[untar\\_files](#)  
[unzip\\_errors](#)  
[unzip\\_files](#)  
[zip\\_errors](#)  
[zip\\_files](#)  
[zipsfx\\_errors](#)  
[zipsfx\\_files](#)

#### 4.4.4.4 ! (Variable)

! returns the last argument of the previous command. The command is retrieved from the history list, so this will not work in a batch file -- it's intended for aliases and command line work.

#### 4.4.4.5 ? variable

If an **external** command (i.e., a program) has an **exit code**, its value is stored in the **?** variable when the program terminates. Additionally, some **internal** commands, e.g., [DIR](#) - to emulate Microsoft's CMD - also set this variable to the same value they set the variable [?](#), an action which destroys the code from the last external command.

To insure that you use the **exit code** from the **external** command you want to check, not that of a subsequent internal or external command, it is best to save the value of **?** in another variable immediately on completion of the external command of interest, and use that variable instead. We also strongly recommend that for internal commands you query the [?](#) variable instead.

Not all programs return an exit code. If a program does not explicitly return an exit code, the value of **%?** is undefined.

**Alternate name:** [ERRORLEVEL](#).

See also: [?](#)

#### 4.4.4.6 \_? variable

**\_?** contains the exit code of the last internal command. You must use or save this value immediately, because it is set by every internal command, including the one used to save it.

**Result codes:**

- 0** command was successful
- 1** a usage error occurred
- 2** another **TCC** error or an operating system error occurred
- 3** the command was interrupted by **Ctrl-C** or **Ctrl-Break**

This variable can also be set in a subroutine by the [RETURN](#) command.

Note that in imitation of CMD some internal commands, e.g., [DIR](#), also set the variables [?](#) and [ERRORLEVEL](#) to the same value they set this variable. However, you are strongly urged to use this variable.

See also: [?](#)

#### 4.4.4.7 = pseudovariable

= is the current [Escape character](#). Use this pseudovariable, instead of the actual escape character, if you want your batch files and aliases to work in other users' environment regardless of how the escape character is defined.

#### 4.4.4.8 + pseudovariable

+ is the current [command separator](#). Use this pseudovariable, instead of the actual command separator, if you want your batch files and aliases to work in other users' environment regardless of how the command separator is defined.

**WARNING:** %+ should always be surrounded by spaces.

For example, if the command separator is an ampersand [&] (the default in **TCC**) both of the commands below will display "Hello" on one line and "world" on the next. However, if the command separator has been changed the first command will display "Hello & echo world", while the second command will

continue to work as intended.

```
echo Hello & echo world
echo Hello %+ echo world
```

#### 4.4.4.9 \_4VER

**\_4VER** returns the current **TCC** version (for example, 16.03). The current [Decimal character](#) is used to separate the major and minor version numbers.

See also: [BUILD](#).

#### 4.4.4.10 \_ACSTATUS

**\_ACSTATUS** returns the AC line status.

| <b>value</b> | <b>meaning</b> |
|--------------|----------------|
| 0            | Offline        |
| 1            | Online         |
| unknown      | Unknown        |

#### 4.4.4.11 \_ADMIN

**\_ADMIN** returns 1 if the current user is an administrator in the local group.

See also [ELEVATED](#).

#### 4.4.4.12 \_AFSWCELL

**\_AFSWCELL** returns the OpenAFS workstation cell.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.4.13 \_ALT

**\_ALT** returns the status of the **Alt** key:

| <b>value</b> | <b>status of selected key</b>     |
|--------------|-----------------------------------|
| 1            | at least one Alt key is depressed |
| 0            | neither is depressed              |

#### 4.4.4.14 \_ANSI

**\_ANSI** returns 1 if the internal **TCC** support for [ANSI Std. X3.64](#) is enabled, or 0 if it is not.

#### 4.4.4.15 \_BATCH

**\_BATCH** returns the current batch file nesting level. It is 0 if no batch file is currently being processed.

Batch files are nested with the internal [CALL](#) command.

#### 4.4.4.16 \_BATCHLINE

**\_BATCHLINE** returns the current line number in the current batch file. It is -1 if no batch file is active.

The first line in the batch file is numbered 1.

**4.4.4.17 \_BATCHNAME**

**\_BATCHNAME** returns the full path and file name of the current batch file. It is an empty string if no batch file is active.

**4.4.4.18 \_BATCHTYPE**

**\_BATCHTYPE** returns the file type of the current batch file:

| <b>value</b> | <b>meaning</b>      |
|--------------|---------------------|
| -1           | not in a batch file |
| 0            | normal              |
| 1            | compressed          |
| 2            | encrypted           |

**4.4.4.19 \_BATTERY**

**\_BATTERY** returns the battery charge status:

| <b>value</b> | <b>meaning</b> |
|--------------|----------------|
| 1            | High           |
| 2            | Low            |
| 4            | Critical       |
| 8            | Charging       |
| 128          | No battery     |
| unknown      | Unknown        |

**4.4.4.20 \_BATTERYLIFE**

**\_BATTERYLIFE** returns either the number of seconds of battery life remaining, or **unknown**.

**4.4.4.21 \_BATTERYPERCENT**

**\_BATTERYPERCENT** returns the percentage of battery charge remaining (0...100), or **unknown**.

**4.4.4.22 \_BDEBUGGER**

**\_BDEBUGGER** returns **1** if the batch debugger is actively debugging a file, or **0** if it is not.

**4.4.4.23 \_BG**

**\_BG** returns a string containing the first three characters of the current background screen output color (for example, **Bla**). See [Colors, Color Names and Codes](#) for details.

**4.4.4.24 \_BOOT**

**\_BOOT** returns the boot drive letter, without a colon.

**4.4.4.25 \_BUILD**

**\_BUILD** returns the internal **TCC** build number.

See also: [\\_4VER](#).

#### 4.4.4.26 \_CAPSLOCK

**\_CAPSLOCK** returns the current state of the *Caps Lock* key on the keyboard:

| <b>value</b> | <b>toggled status</b> |
|--------------|-----------------------|
| 1            | ON                    |
| 0            | OFF                   |

#### 4.4.4.27 \_CDROMS

**\_CDROMS** returns a space-delimited list of the CD-ROM drives on the system.

#### 4.4.4.28 \_CHILDPID

**\_CHILDPID** returns the process ID of the most recent child process.

#### 4.4.4.29 \_CI

**\_CI** returns the insert mode cursor shape, as a percentage (**0** to **100**).

See also [SETDOS /S](#) and the [Insert Cursor](#) configuration option.

#### 4.4.4.30 \_CMDLINE

**\_CMDLINE** returns the current command line. (This is most useful in key aliases.) If you specify it on the command line, it is expanded to the contents of the command line (not including the %\_cmdline variable itself).

**Example:**

```
echo one two three %_cmdline
```

will return:

```
one two three echo one two three
```

#### 4.4.4.31 \_CMDPROC

**\_CMDPROC** returns the name of the current command processor (**TCC** or **TCCL**).

#### 4.4.4.32 \_CMDSPEC

**\_CMDSPEC** returns the full pathname of the command processor.

#### 4.4.4.33 \_CO

**\_CO** returns the overstrike mode cursor shape, as a percentage (**0** to **100**).

See also [SETDOS /S](#) and the [Overstrike Cursor](#) configuration option.

#### 4.4.4.34 \_CODEPAGE

**\_CODEPAGE** returns the input code page used by the TCC console.

See also [CHCP](#).

**4.4.4.35 \_COLUMN**

**\_COLUMN** is the current cursor column. The leftmost column is numbered **0**.

See also [\\_COLUMNS](#), [\\_ROW](#), and [\\_ROWS](#).

**4.4.4.36 \_COLUMNS**

**\_COLUMNS** returns the current number of virtual screen columns (for example, **80**).

See [Resizing the Take Command Window](#) for additional details on the virtual screen width.

See also [\\_COLUMN](#), [\\_ROW](#), and [\\_ROWS](#).

**4.4.4.37 \_CONSOLEB**

**\_CONSOLEB** returns the handle to the active console screen buffer.

See also [@CONSOLEB](#).

**4.4.4.38 \_CONSOLEPIDS**

**\_CONSOLEPIDS** returns a space-delimited list of the process IDs of all processes attached to this console.

**4.4.4.39 \_COUNTRY**

**\_COUNTRY** returns the current country code as reported by the operating system. This code is usually the same as the international dialing code for the country.

**4.4.4.40 \_CPU**

**\_CPU** returns the CPU type:

486 i486  
586 Pentium family  
etc.

This variable merely queries Windows for the processor type. Compatible AMD or other processors will generally return the value corresponding to the Intel processor they most closely resemble.

**\_CPU** is obsolete. To determine the CPU type, revision, stepping level, and other details, use the [@WININFO](#) or [@WMI](#) functions.

**4.4.4.41 \_CPUUSAGE**

**\_CPUUSAGE** returns the current CPU usage, as a percent (**0** to **100**).

**4.4.4.42 \_CTRL**

**\_CTRL** returns the status of the **Ctrl** keys:

| <b>value</b> | <b>status of selected key</b>      |
|--------------|------------------------------------|
| 1            | at least one Ctrl key is depressed |
| 0            | neither is depressed               |

#### 4.4.4.43 **\_CWD**

**\_CWD** returns the current working directory, in the format ***d:\ pathname***. If the current working directory is a root directory, the format is ***d:\***.

See also [\\_CWDS](#), [\\_CWP](#), [\\_CWPS](#), [@CWD](#), and [@CWDS](#).

#### 4.4.4.44 **\_CWDS**

**\_CWDS** returns the current working directory in the format ***d:\ pathname\***.

See also [\\_CWD](#), [\\_CWP](#), [\\_CWPS](#), [@CWD](#), and [@CWDS](#).

#### 4.4.4.45 **\_CWP**

**\_CWP** returns the current working directory in the format ***\ pathname*** (without the drive letter).

See also [\\_CWD](#), [\\_CWDS](#), [\\_CWPS](#), [@CWD](#), and [@CWDS](#).

#### 4.4.4.46 **\_CWPS**

**\_CWPS** returns the current working directory in the format ***\ pathname\*** (without the drive letter).

See also [\\_CWD](#), [\\_CWDS](#), [\\_CWP](#), [@CWD](#), and [@CWDS](#).

#### 4.4.4.47 **\_DATE**

**\_DATE** returns the current system date, in the format determined by your country settings. The year will be in two-digit format for compatibility unless your country setting is ***yyyy-mm-dd***.

See also [\\_ISODATE](#).

#### 4.4.4.48 **\_DATETIME**

**\_DATETIME** returns the current date and time in the format ***yyyyMMddhhmmss***. The date part is the same as [\\_isodate](#) without separators.

For the current UTC time, see [\\_UTCDATETIME](#).

#### 4.4.4.49 **\_DAY**

**\_DAY** returns the current day of the month (1 to 31).

#### 4.4.4.50 **\_DETACHPID**

**\_DETACHPID** returns the process ID of the most recent process launched by the [DETACH](#) command.

#### 4.4.4.51 **\_DISK**

**\_DISK** returns the current disk drive letter, without a colon (for example, **C**).

If the current directory is a UNC, **%\_disk** will return the share name.

#### 4.4.4.52 **\_DNAME**

**\_DNAME** returns the name of the file used to store file descriptions. It can be changed with the [Description Filename](#) configuration option, or the [SETDOS /D](#) command.

**4.4.4.53 \_DOS**

**\_DOS** returns the operating system type. **Take Command** returns a different value depending on the operating system, as follows:

| <i>Platform</i>     | <i>Take Command</i> |
|---------------------|---------------------|
| Windows XP          | WINXP               |
| Windows 2003        | WIN2003             |
| Windows Vista       | WINVISTA            |
| Windows 2008 Server | WIN2008             |
| Windows 7           | WINDOWS7            |
| Windows 8           | WINDOWS8            |
| Windows 2012 Server | WIN2012             |

This variable is useful if you have batch files running in more than one environment, and need to take different actions depending on the underlying operating environment or command processor. See also the [\\_WINVER](#) variable.

**4.4.4.54 \_DOSVER**

**\_DOSVER** returns the current operating system version. The current [Decimal character](#) is used to separate the major and minor version numbers.

**4.4.4.55 \_DOW**

**\_DOW** returns the first three characters of the name of the current day of the week (**Mon**, **Tue**, **Wed**, etc.).

**\_DOW** returns the English name for the day of the week. For a localized version, see [\\_IDOW](#).

**4.4.4.56 \_DOWF**

**\_DOWF** returns the full name of the day of the week for the current date (**Monday**, **Tuesday**, etc.).

**\_DOWF** returns the English name for the day of the week. For a localized version, see [\\_IDOWF](#).

**4.4.4.57 \_DOWI**

**\_DOWI** returns the current day of the week as an integer (**1** = Sunday, **2** = Monday, etc.).

**4.4.4.58 \_DOY**

**\_DOY** returns the current day of the year (1 to 366).

**4.4.4.59 \_DRIVES**

**\_DRIVES** returns a space-delimited list of the existing drives in the format:

A: C: D: E:

**\_DRIVES** only checks to see if the drive exists, not whether it is ready.

**4.4.4.60 \_DST**

**\_DST** returns 1 if daylight savings time is in effect, or 0 if it is not.

#### 4.4.4.61 \_DVDS

`_DVDS` returns a space-delimited list of the DVD drives on the system.

#### 4.4.4.62 \_ECHO

`_ECHO` returns the current echo state (**0**=off, **1**=on). There are two ECHO states, one for the command line and one for batch files (see the [ECHO](#) command and the [Batch Echo](#) configuration option). The value returned by the `_ECHO` variable reflects the state applicable at the time the variable is queried.

#### 4.4.4.63 \_EDITMODE

`_EDITMODE` returns 0 if the line editor is in overstrike mode, or 1 if it is in insert mode.

#### 4.4.4.64 \_ELEVATED

`_ELEVATED` returns 1 if the **TCC** process is elevated. (Windows Vista and later only.)

#### 4.4.4.65 \_EXECARRAY

`_EXECARRAY` returns the number of array elements assigned by the last [@EXECARRAY](#) function.

#### 4.4.4.66 \_EXECSTR

`_EXECSTR` returns the integer return code of the last [@EXECSTR](#) function.

#### 4.4.4.67 \_EXIT

`_EXIT` returns the reason for exiting **TCC**. The possible values are:

- 0** EXIT command
- 2** CLOSE\_EVENT
- 5** LOGOFF\_EVENT
- 6** SHUTDOWN\_EVENT

#### 4.4.4.68 \_EXPANSION

`_EXPANSION` returns the current expansion mode (i.e., [SETDOS /X](#)). It returns the string **0** if everything is enabled, or a string of up to 9 characters of the disabled modes.

For example, if you disable nested variable expansion and redirection:

```
setdos /x-46
```

then `%_expansion` will return **46**.

#### 4.4.4.69 \_FG

`_FG` returns a string containing the first three letters of the current foreground screen output color (for example, "Whi"). See [Colors, Color Names and Codes](#) for details.

#### 4.4.4.70 \_FTPERROR

`_FTPERROR` returns the error code of the last error reported by [FTP](#). Some of the possible codes are:

- 101** You cannot change the remote host at this time
- 102** The remote host address is invalid
- 118** Firewall error
- 141** FTP protocol error

|              |                                                  |
|--------------|--------------------------------------------------|
| <b>142</b>   | Communication error                              |
| <b>143</b>   | Busy performing current action                   |
| <b>144</b>   | Local file error                                 |
| <b>145</b>   | Can't open local file for reading                |
| <b>146</b>   | No remote file specified while uploading         |
| <b>147</b>   | Data interface error                             |
| <b>301</b>   | Operation interrupted                            |
| <b>302</b>   | Can't open local file                            |
| <b>311</b>   | Accept failed for data connection                |
| <b>312</b>   | Asynchronous select failed for data connection   |
| <b>11001</b> | Host not found                                   |
| <b>11002</b> | Non-authoritative 'Host not found'               |
| <b>11003</b> | Non-recoverable errors: FORMERR, REFUSED, NOTIMP |
| <b>11104</b> | Valid name, no data record (check DNS setup)     |

#### 4.4.4.71 \_HDRIVES

\_HDRIVES returns a space-delimited list of the hard (fixed) drives on the system.

#### 4.4.4.72 \_HLOGFILE

\_HLOGFILE returns the name of the current history log file (or an empty string if LOG /H is OFF). See [LOG](#) for information on history logging.

#### 4.4.4.73 \_HOST

\_HOST returns the host name for the local computer.

#### 4.4.4.74 \_HOUR

\_HOUR returns the current hour (0 - 23) in local time.

For the current UTC time, see [UTCHOUR](#).

#### 4.4.4.75 \_HWP PROFILE

\_HWP PROFILE returns the name of the current Windows hardware profile.

#### 4.4.4.76 \_IDE

\_IDE returns 1 when in the IDE / batch debugger, or 0 if not.

See also [BDEBUGGER](#).

#### 4.4.4.77 \_IDLETICKS

\_IDLETICKS returns the number of milliseconds since the last user input.

#### 4.4.4.78 \_IDOW

\_IDOW returns the 3-character abbreviation for the day of the week for the current date, in the current locale language.

See [DOW](#) for the English language only version.

#### 4.4.4.79 \_IDOWF

\_IDOWF returns the full name for the day of the week for the current date, in the current locale language.

For the English language only version, see [\\_DOWF](#).

#### **4.4.4.80 \_IFTP**

**\_IFTP** returns 1 if an [IFTP](#) session is active, 0 if it is not.

#### **4.4.4.81 \_IFTPS**

**\_IFTPS** returns 1 if an SSL [IFTP](#) session is active, 0 if it is not.

#### **4.4.4.82 \_IMONTH**

**\_IMONTH** returns the abbreviated name for the current month, in the current locale language.

#### **4.4.4.83 \_IMONTHF**

**\_IMONTHF** returns the full name for the current month, in the current locale language.

#### **4.4.4.84 \_ININAME**

**\_ININAME** returns the fully qualified pathname of the INI file used by the current shell.

#### **4.4.4.85 \_INSERT**

**\_INSERT** returns 0 if the line editor is currently in overstrike mode, or 1 if it is in insert mode.

See also [\\_EDITMODE](#).

#### **4.4.4.86 \_IP**

**\_IP** returns the IP address of the local computer. If the computer has more than one NIC, **\_IP** returns a space-delimited list of all IP addresses.

#### **4.4.4.87 \_IPADAPTER**

**\_IPADAPTER** returns the index of the current adapter.

#### **4.4.4.88 \_IPADAPTERS**

**\_IPADAPTERS** returns the number of adapters in the system.

#### **4.4.4.89 \_IPARPPROXY**

**\_IPARPPROXY** returns 1 if the local computer is acting as an ARP proxy, or 0 if it is not.

#### **4.4.4.90 \_IPDNS**

**\_IPDNS** returns 1 if DNS is enabled for the local computer, or 0 if it is not enabled.

#### **4.4.4.91 \_IPDNSSERVER**

**\_IPDNSSERVER** returns the default DNS server for the local computer.

**4.4.4.92 \_IPROUTING**

**\_IPROUTING** returns 1 if routing is enabled on the local computer, or 0 if it is not.

**4.4.4.93 \_ISFTP**

**\_ISFTP** returns 1 if an SSH [IFTP](#) session is active, 0 if it is not.

**4.4.4.94 \_ISODATE**

**\_ISODATE** returns the current local system date, in ISO 8601 format (**yyyy-mm-dd**).

See also [DATE](#) and [DATETIME](#).

**4.4.4.95 \_ISODOWI**

**\_ISODOWI** returns the ISO 8601 numeric day of the week (Monday=1, Sunday=7).

**4.4.4.96 \_ISOWDATE**

**\_ISOWDATE** returns the ISO 8601 current week date (yyyy-Www-d).

**4.4.4.97 \_ISOWEEK**

**\_ISOWEEK** returns the ISO 8601 week of year.

**4.4.4.98 \_ISOWYEAR**

**\_ISOWYEAR** returns the ISO 8601 week date year.

**4.4.4.99 \_KBHIT**

**\_KBHIT** returns 1 if one or more keystrokes are waiting in the keyboard buffer, or 0 if the keyboard buffer is empty.

**4.4.4.100 \_LALT**

**\_LALT** returns the status of the left **Alt** key on the keyboard:

| <b>value</b> | <b>key status</b> |
|--------------|-------------------|
| 1            | depressed         |
| 0            | not depressed     |

See also [ALT](#) and [RALT](#).

**4.4.4.101 \_LASTDIR**

**\_LASTDIR** returns the previous directory (from the directory history).

**4.4.4.102 \_LASTDISK**

**\_LASTDISK** returns the last valid drive letter (without a colon).

**4.4.4.103 \_LCTRL**

**\_LCTRL** returns the status of the Left Ctrl key on the keyboard:

| <b>value</b> | <b>key status</b> |
|--------------|-------------------|
| 1            | depressed         |
| 0            | not depressed     |

See also [CTRL](#) and [RCTRL](#).

#### 4.4.4.104 \_LOGFILE

**\_LOGFILE** returns the name of the current log file (or an empty string if LOG is OFF). See [LOG](#) for information on logging.

#### 4.4.4.105 \_LSHIFT

**\_LSHIFT** returns the status of the left shift key on the keyboard:

| <b>value</b> | <b>key status</b> |
|--------------|-------------------|
| 1            | depressed         |
| 0            | not depressed     |

See also [SHIFT](#) and [RSHIFT](#).

#### 4.4.4.106 \_MINUTE

**\_MINUTE** returns the current minute (0 - 59).

For the current UTC time, see [UTCMINUTE](#).

#### 4.4.4.107 \_MONITORS

**\_MONITORS** returns the number of video displays.

#### 4.4.4.108 \_MONTH

**\_MONTH** returns the current numeric month of the year (1 to 12).

#### 4.4.4.109 \_MONTHF

**\_MONTHF** returns the full name of the current month (**January**, **February**, etc.).

#### 4.4.4.110 \_NUMLOCK

**\_NUMLOCK** reports the current state of the **Num Lock** key:

| <b>value</b> | <b>toggled status</b> |
|--------------|-----------------------|
| 1            | ON                    |
| 0            | OFF                   |

#### 4.4.4.111 \_OPENAFS

**\_OPENAFS** returns **1** if the [OpenAFS](#) service is active, **0** if it is not.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.4.112 \_OSBUILD

**\_OSBUILD** returns the Windows build number. The build number does not include the major or minor version.

#### 4.4.4.113 \_PARENT

**\_PARENT** returns the name of the parent process (the process that started **TCC**).

**4.4.4.114 \_PID**

**\_PID** returns the process ID number for the current **TCC** process.

**4.4.4.115 \_PIPE**

**\_PIPE** returns **1** if the current process is running inside a pipe, and **0** otherwise.

**4.4.4.116 \_PPID**

**\_PPID** returns the process ID number of the parent process.

**4.4.4.117 \_RALT**

**\_RALT** returns the status of the right Alt key on the keyboard:

| <b>value</b> | <b>key status</b> |
|--------------|-------------------|
| 1            | depressed         |
| 0            | not depressed     |

See also [\\_ALT](#) and [\\_LALT](#).

**4.4.4.118 \_RCTRL**

**\_RCTRL** returns the status of the right Ctrl key on the keyboard:

| <b>value</b> | <b>key status</b> |
|--------------|-------------------|
| 1            | depressed         |
| 0            | not depressed     |

See also [\\_CTRL](#) and [\\_LCTRL](#).

**4.4.4.119 \_READY**

**\_READY** returns a space-delimited list of the currently ready (accessible) drives in the format :

C: D: E:

**4.4.4.120 \_REGISTERED**

**\_REGISTERED** returns the registered name of the user or an empty string if **Take Command** isn't registered.

**4.4.4.121 \_ROW**

**\_ROW** returns the current cursor row (for example, **0** for the top of the window).

**4.4.4.122 \_ROWS**

**\_ROWS** returns the current number of screen rows in the **TCC** window (for example, **25**).

**4.4.4.123 \_RSHIFT**

**\_RSHIFT** returns the status of the right Shift key on the keyboard:

| <b>value</b> | <b>key status</b> |
|--------------|-------------------|
| 1            | depressed         |
| 0            | not depressed     |

See also [SHIFT](#) and [LSHIFT](#).

#### 4.4.4.124 \_RUBYTYPE

**\_RUBYTYPE** returns the type of the Ruby VALUE returned by the last [@RUBY](#) call.

#### 4.4.4.125 \_RUBYVALUE

**\_RUBYVALUE** returns the Ruby VALUE returned by the last [@RUBY](#) call.

#### 4.4.4.126 \_SCROLLLOCK

**\_SCROLLLOCK** reports the current *scroll lock* state, which can be toggled using the *scroll lock* key:

| <b>value</b> | <b>toggled status</b> |
|--------------|-----------------------|
| 1            | ON                    |
| 0            | OFF                   |

#### 4.4.4.127 \_SECOND

**\_SECOND** is the current second (0 - 59).

For the current UTC time, see [UTCSECOND](#).

#### 4.4.4.128 \_SELECTED

**\_SELECTED** returns the first line of text highlighted in the **Take Command** tab window. If no text has been highlighted, **\_SELECTED** returns an empty string.

#### 4.4.4.129 \_SERIALPORTS

**\_SERIALPORTS** returns a space-delimited list of all of the available serial ports (COM1 - COMn). If there are no serial ports on the system, **\_SERIALPORTS** returns an empty string.

#### 4.4.4.130 \_SERVICE

**\_SERVICE** returns 1 if **TCC** was started as a service (TCC /N).

#### 4.4.4.131 \_SHELL

**\_SHELL** is the current shell number. **\_SHELL** will return 0 for a primary shell, or 1 (or higher) for a **TCC** shell instance started by a parent **TCC** process (either directly or via a pipe).

Note that the concept of shell numbers is now mostly obsolete in Windows.

#### 4.4.4.132 \_SHIFT

**\_SHIFT** is the status the two **Shift** keys:

| <b>value</b> | <b>status of selected key</b> |
|--------------|-------------------------------|
| 1            | at least one is depressed     |
| 0            | neither is depressed          |

#### 4.4.4.133 \_SHORTCUT

**\_SHORTCUT** returns the full pathname of the shortcut file that started this process. If the process was not started from a shortcut, **\_SHORTCUT** returns an empty string.

#### 4.4.4.134 \_SHRALIAS

**\_SHRALIAS** returns **1** if [SHRALIAS](#) is loaded, **0** if it is not.

#### 4.4.4.135 \_STARTPATH

**\_STARTPATH** returns the startup directory for the current **TCC** shell. (This is not necessarily the same as the location of the **TCC** executable!)

#### 4.4.4.136 \_STARTPID

**\_STARTPID** returns the process ID of the most recent process launched by the [START](#) command.

#### 4.4.4.137 \_STDIN

**\_STDIN** returns **1** if STDIN points to the console, or **0** if it has been redirected.

#### 4.4.4.138 \_STDOUT

**\_STDOUT** returns **1** if STDOUT points to the console, or **0** if it has been redirected.

#### 4.4.4.139 \_STDERR

**\_STDERR** returns **1** if STDERR points to the console, or **0** if it has been redirected.

#### 4.4.4.140 \_STZN

**\_STZN** returns the name of standard time in the current time zone.

See also [STZO](#), [TZN](#), and [TZO](#).

#### 4.4.4.141 \_STZO

**\_STZO** returns the offset in minutes from UTC for standard time in the current time zone.

See also [STZN](#), [TZN](#), and [TZO](#).

#### 4.4.4.142 \_SYSERR

**\_SYSERR** returns the error code of the last Windows system error.

See the [Windows System Errors](#) table in the Reference section for examples.

#### 4.4.4.143 \_TCCVER

**\_TCCVER** returns the current **TCC** version (for example, 16.03). The current [Decimal character](#) is used to separate the major and minor version numbers.

See also: [BUILD](#).

#### 4.4.4.144 \_TCEEXIT

`_TCEEXIT` returns the full pathname of the TCEEXIT.\* file, or an empty string if **TCC** can't find TCEEXIT.

Note that the string returned by `_TCEEXIT` can change before TCEEXIT is actually executed. (For example, if you modify the TCMD.INI settings.)

#### 4.4.4.145 \_TCFILTER

`_TCFILTER` returns the current filter in the **Take Command** List view window if **TCC** is running in a tab window, or an empty string if it is not.

#### 4.4.4.146 \_TCFOLDER

`_TCFOLDER` returns the selected folder in the **Take Command** Folders window if **TCC** is running in a tab window, or an empty string if it is not.

#### 4.4.4.147 \_TCLISTVIEW

`_TCLISTVIEW` returns the selected entries in the **Take Command** List View window if **TCC** is running in a tab window, or an empty string if it is not.

#### 4.4.4.148 \_TCSTART

`_TCSTART` returns the full pathname of the TCSTART.\* file, or an empty string if **TCC** didn't find TCSTART.

#### 4.4.4.149 \_TCTAB

`_TCTAB` returns **1** if this **TCC** process is running in a **Take Command** tab window, or **0** if it is not.

#### 4.4.4.150 \_TCTABACTIVE

`_TCTABACTIVE` returns **1** if this **TCC** instance is the active tab in **Take Command**.

#### 4.4.4.151 \_TCTABS

`_TCTABS` returns the current number of **Take Command** tab windows (or **0** if **TCC** is not running in **Take Command**).

#### 4.4.4.152 \_TIME

`_TIME` returns the current system time in the format **hh:mm:ss**. The separator character may vary depending upon your country information.

#### 4.4.4.153 \_TRANSIENT

`_TRANSIENT` returns **1** if the current shell is transient (started with a **/C**, see [Command Line Options](#) for details), or **0** otherwise.

#### 4.4.4.154 \_TZN

`_TZN` returns the name of the current time zone.

See also [\\_STZN](#), [\\_STZO](#), and [\\_TZO](#).

**4.4.4.155 \_TZO**

**\_TZO** returns the offset in minutes from UTC for the current time zone.

See also [\\_STZN](#), [\\_STZO](#), and [\\_TZN](#).

**4.4.4.156 \_UNICODE**

**\_UNICODE** returns **1** if the shell is currently using Unicode for redirected output, **0** otherwise.

**4.4.4.157 \_UTCDATE**

**\_UTCDATE** returns the current UTC date in the user's default format.

**4.4.4.158 \_UTCDATETIME**

**\_UTCDATETIME** returns the current UTC date and time.

For the local time, see [\\_DATETIME](#).

**4.4.4.159 \_UTCHOUR**

**\_UTCHOUR** returns the current UTC hour.

For the local time, see [\\_HOUR](#).

**4.4.4.160 \_UTCISODATE**

**\_UTCISODATE** returns the current UTC date in ISO format (yyyy-mm-dd).

For the current local date, see [\\_ISODATE](#).

**4.4.4.161 \_UTCMINUTE**

**\_UTCMINUTE** returns the current UTC minute.

For the current local time, see [\\_UTCMINUTE](#).

**4.4.4.162 \_UTCSECOND**

**\_UTCSECOND** returns the current UTC second.

For the current local time, see [\\_SECOND](#).

**4.4.4.163 \_UTCTIME**

**\_UTCTIME** returns the current UTC time.

See [\\_TIME](#) to retrieve the current local time.

**4.4.4.164 \_VERMAJOR**

**\_VERMAJOR** returns the **TCC** major version number (i.e., **10**).

**4.4.4.165 \_VERMINOR**

**\_VERMINOR** returns the **TCC** minor version number (the tenths or hundredths digit).

For example, for **Take Command** 16.01, **\_VERMINOR** will return **1**.

**4.4.4.166 \_VERSION**

**\_VERSION** returns the **TCC** version in *major.minor* format (i.e., **16.0**).

**4.4.4.167 \_VIRTUALBOX**

**\_VIRTUALBOX** returns 1 if **TCC** is running inside a VirtualBox virtual machine.

**4.4.4.168 \_VIRTUALPC**

**\_VIRTUALPC** returns 1 if **TCC** is running inside a VirtualPC virtual machine. (Not supported in x64 TCC.)

**4.4.4.169 \_VMWARE**

**\_VMWARE** returns 1 if **TCC** is running inside a VMWare virtual machine. (Not supported in x64 TCC.)

**4.4.4.170 \_VOLUME**

**\_VOLUME** returns the current volume level of the default audio device.

**4.4.4.171 \_VXPIXELS**

**\_VXPIXELS** returns the horizontal size of the virtual screen (including multiple monitors) in pixels.

**4.4.4.172 \_VYPIXELS**

**\_VYPIXELS** returns the vertical size of the virtual screen (including multiple monitors) in pixels.

**4.4.4.173 \_WINDIR**

**\_WINDIR** returns the pathname of the Windows directory.

**4.4.4.174 \_WINFGWINDOW**

**\_WINFGWINDOW** returns the title of the foreground window.

**4.4.4.175 \_WINNAME**

**\_WINNAME** returns the computer name of the current system.

**4.4.4.176 \_WINSYSDIR**

**\_WINSYSDIR** returns the pathname of the Windows system directory.

**4.4.4.177 \_WINTICKS**

**\_WINTICKS** returns the number of milliseconds since Windows was started.

**4.4.4.178 \_WINTITLE**

**\_WINTITLE** returns the title of the current window.

**4.4.4.179 \_WINUSER**

**\_WINUSER** returns the name of the user currently logged on.

**4.4.4.180 \_WINVER**

**\_WINVER** returns the current Windows version number. The current [Decimal character](#) is used to separate the major and minor version numbers.

**4.4.4.181 \_WOW64**

**\_WOW64** returns 1 if **TCC** is running in the WOW64 environment (64-bit Windows). Note that this only applies to the 32-bit version of **TCC**. If you want to know if you're running **TCC** x64, see [%\\_X64](#).

**4.4.4.182 \_WOW64DIR**

**\_WOW64DIR** returns the system Wow64 directory (x64 Windows only).

**4.4.4.183 \_X64**

**\_X64** returns 1 if **TCC** is the x64 (64-bit) version running on a 64-bit version of Windows.

**4.4.4.184 \_XMOUSE**

**\_XMOUSE** returns the column position of the most recent left mouse click. (Note that this will only work in a **Take Command** tab window, or if you have enabled the console mouse in a stand-alone **TCC** session.)

**4.4.4.185 \_XPIXELS**

**\_XPIXELS** returns the number of horizontal pixels on the current physical display.

See also [\\_YPIXELS](#).

**4.4.4.186 \_XWINDOW**

**\_XWINDOW** returns the width of the **Take Command** or **TCC** window in pixels.

**4.4.4.187 \_YEAR**

**\_YEAR** returns the current year (1980 to 2099).

**4.4.4.188 \_YMOUSE**

**\_YMOUSE** returns the row position of the most recent left mouse click. (Note that this will only work in a **Take Command** tab window, or if you have enabled the console mouse in a stand-alone **TCC** session.)

**4.4.4.189 \_YPIXELS**

**\_YPIXELS** returns the number of vertical pixels on the current physical display.

See also [\\_XPIXELS](#).

**4.4.4.190 \_YWINDOW**

**\_YWINDOW** returns the height of the **Take Command** or **TCC** window in pixels.

**4.4.4.191 ERRORLEVEL**

**ERRORLEVEL** is an alternate name (included for compatibility with CMD) for the [?](#) variable, and is the exit code of the last external command. Many programs return **0** to indicate success and a non-zero value to signal an error. However, not all programs return an exit code. If no explicit exit code is returned, the value of **ERRORLEVEL** is undefined.

**WARNING:** For compatibility with CMD, some internal commands, e.g., **DIR**, also set this variable to the same value as the variable [?](#), which destroys the code from the last external command. If you need to preserve the return value of the external command, save the value in a variable immediately upon command completion, and use the saved variable instead. We also strongly recommend that for internal commands you query the [?](#) variable instead.

See also [?](#)

#### 4.4.5 Functions

**Variable functions** are very similar to internal variables, but they take one or more parameters (which can be environment variables or even other variable functions).

Variable functions are useful at the command prompt as well as in [aliases](#) and [batch files](#) to check on available system resources, manipulate strings and numbers, and work with files and filenames.

The variable functions built into **TCC** are listed in alphabetical order in subsequent topics. You can also obtain help from the command prompt on any function with a **HELP @functionname** command, or by pressing [Ctrl-F1](#) when the cursor is on the function name. See the [HELP](#) command for details

**Note:** The [FUNCTION](#) command can be used to create, edit, or display user-defined variable functions, and the [UNFUNCTION](#) to delete them.

For a list of Variable Functions organized by general categories of use, see [Variable Functions by Category](#).

#### Syntax

To have either a user-defined or a built-in variable function evaluated, its name must be preceded by a percent sign % (%@**EVAL**, %@**LEN**, etc.). All variable functions must have square brackets [] enclosing their parameter(s), if any. No space is allowed between the function name and the [. The combined parameters of a variable function may not exceed 32,767 characters.

#### Memory Size / Disk Space / File Size Units and Report Format

Some variable functions, such as [@DISKFREE](#), accept an optional parameter **scale code**. These functions return a size of a disk or of an entity on the disk as a multiple of the specified scale factor from the table below. Lower case letters denote a power of 1,000, upper case letters denote a power of 1,024.

| Code     | Scale Factor |                           | Code     | Scale Factor |                           | Unit Name |
|----------|--------------|---------------------------|----------|--------------|---------------------------|-----------|
| <b>b</b> |              | 1                         | <b>B</b> |              | 1                         | byte      |
| <b>k</b> |              | 1,000                     | <b>K</b> |              | 1,024                     | kilobyte  |
| <b>m</b> |              | 1,000,000                 | <b>M</b> |              | 1,048,576                 | megabyte  |
| <b>g</b> |              | 1,000,000,000             | <b>G</b> |              | 1,073,741,824             | gigabyte  |
| <b>t</b> |              | 1,000,000,000,000         | <b>T</b> |              | 1,099,511,627,776         | terabyte  |
| <b>p</b> |              | 1,000,000,000,000,000     | <b>P</b> |              | 1,125,899,906,842,624     | petabyte  |
| <b>e</b> |              | 1,000,000,000,000,000,000 | <b>E</b> |              | 1,152,921,504,606,846,976 | exabyte   |

You can include **commas** (or the [thousands separator](#)) in the value returned from a function by appending the letter **c** to the scale code. For example, to add commas to a **b** (number of bytes) result, enter **bc** as the parameter, i.e.:

```
echo %@DISKFREE[C, bc]
```

#### Notes

- 1) Disk manufacturers use the prefixes adopted from the metric system (kilo, mega, giga, tera) in their original meaning (powers of 1,000), while memory manufacturers and Microsoft use the slightly larger powers of 1,024 ( $2^{10}$ ).
- 2) The **scale code** is one of the few instances in which **TCC** is case sensitive.

### Date Parameter Format

See the [Date Formats](#) topic.

### File Name Parameters

Filenames passed as variable function parameters must be enclosed in double quotes if they contain white space or special characters. Several functions also return filenames or parts of filenames. On LFN drives, the strings returned by these functions may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. For example (either of these methods would work):

```
set fname="%@findfirst[pro*]"
echo First PRO file contains:
type %fname
.....
set fname=%@findfirst[pro*]
echo First PRO file contains:
type "%fname"
.....
```

If you don't use the quotes in the SET or TYPE command in this example, TYPE will not interpret white space or special characters in the name properly.

### Drive Letter Parameters

In variable functions which take a drive letter as a parameter, like [@DISKFREE](#) or [@READY](#), the drive letter must be followed by a colon. The function will not work properly if you use the drive letter without the colon.

### Functions Accessing File Handles

The [@FILEREAD](#), [@FILEWRITE](#), [@FILEWRITEB](#), [@FILESEEK](#), [@FILESEEKL](#), and [@FILECLOSE](#) functions allow you to access files based on their file handle. These functions must be used only with file handles returned by [@FILEOPEN](#), unless otherwise noted under the individual functions. If you use them with any other file handle you may damage files.

### File Attributes

Several functions accept a file attribute string to help determine which files to process. The rules for constructing the attribute string are the same as the ones for [Attribute Switches](#) in commands.

### Examples

You can use variable functions in a wide variety of ways depending on your needs. Here are a couple of examples to give you an idea of what's possible:

The command below sets the prompt to show the amount of free memory (see [PROMPT](#) for details on including variable functions in your prompt):

```
prompt (%{@dosmem[K]K}) pg
```

Set up a simple command line calculator. The calculator is used with a command such as CALC 3 \* (4 + 5):

```
alias calc `echo The answer is: %@eval[%$]`
```

#### 4.4.5.1 Functions by Name

See also [Functions by Category](#).

|                             |                                         |
|-----------------------------|-----------------------------------------|
| <a href="#">@ABS</a>        | Absolute value of number                |
| <a href="#">@AFSCELL</a>    | OpenAFS cell name                       |
| <a href="#">@AFSMOUNT</a>   | OpenAFS mount point                     |
| <a href="#">@AFSPATH</a>    | Path in OpenAFS: 1, otherwise 0         |
| <a href="#">@AFSSYMLINK</a> | OpenAFS symbolic link                   |
| <a href="#">@AFSVOLID</a>   | OpenAFS volume ID                       |
| <a href="#">@AFSVOLNAME</a> | OpenAFS volume name                     |
| <a href="#">@AGEDATE</a>    | Converts an age into date and time      |
| <a href="#">@ALIAS</a>      | Value of an alias                       |
| <a href="#">@ALTNAMES</a>   | Short name for the file                 |
| <a href="#">@ASCII</a>      | Set of ASCII-s for characters in string |
| <a href="#">@ARRAYINFO</a>  | Array variable information              |
| <a href="#">@ASSOC</a>      | File association                        |
| <a href="#">@ATTRIB</a>     | Test or return file attributes          |
| <a href="#">@AVERAGE</a>    | Average of a list of numbers            |

|                            |                                      |
|----------------------------|--------------------------------------|
| <a href="#">@B64DECODE</a> | Decode Base64 file or string         |
| <a href="#">@B64ENCODE</a> | Encode file or string as Base64      |
| <a href="#">@ALLOC</a>     | Allocate a binary buffer             |
| <a href="#">@BFREE</a>     | Free a binary buffer                 |
| <a href="#">@BPEEK</a>     | Read a value from a binary buffer    |
| <a href="#">@BPEEKSTR</a>  | Read a string from a binary buffer   |
| <a href="#">@BPOKE</a>     | Write a value to a binary buffer     |
| <a href="#">@BPOKESTR</a>  | Write a string to a binary buffer    |
| <a href="#">@BREAD</a>     | Read from a file to a binary buffer  |
| <a href="#">@BWRITE</a>    | Write from a binary buffer to a file |

|                          |                                                    |
|--------------------------|----------------------------------------------------|
| <a href="#">@CAPI</a>    | Call a _cdecl function in a DLL                    |
| <a href="#">@CAPS</a>    | Capitalize first character of each word            |
| <a href="#">@CDROM</a>   | CD-ROM drive: 1, otherwise 0                       |
| <a href="#">@CEILING</a> | Smallest integer not less than a number            |
| <a href="#">@CHAR</a>    | Character string, given a set of ASCII-s           |
| <a href="#">@CLIP</a>    | Specified line from clipboard                      |
| <a href="#">@CLIPW</a>   | Write string to the clipboard                      |
| <a href="#">@COLOR</a>   | RGB value of a color                               |
| <a href="#">@COMMA</a>   | Insert commas into a number (thousands separators) |

|           |                                                                       |
|-----------|-----------------------------------------------------------------------|
| @COMPARE  | Two files are identical: 1, otherwise 0                               |
| @CONSOLE  | Identify console sessions                                             |
| @CONSOLEB | Create or restore a console screen buffer                             |
| @CONVERT  | Convert value from input base to output base                          |
| @COUNT    | Number of times a character appears in a string                       |
| @CRC32    | File CRC                                                              |
| @CWD      | Current Working Directory of specified drive                          |
| @CWDS     | Current Working Directory of specified drive, with trailing backslash |

|              |                                      |
|--------------|--------------------------------------|
| @DATE        | Convert date to number of days       |
| @DATECONV    | Convert date to another format       |
| @DAY         | Day of month for date                |
| @DEBUG       | Write a string to the debugger       |
| @DEC         | Decrement a numeric value by 1       |
| @DECIMAL     | Decimal portion of a number          |
| @DESCRIPT    | File description                     |
| @DEVICE      | Character device: 1, otherwise 0     |
| @DIGITS      | String is all digits: 1, otherwise 0 |
| @DIRSTACK    | Directory stack entry                |
| @DISKFREE    | Free disk space                      |
| @DISKTOTAL   | Total disk space                     |
| @DISKUSED    | Used disk space                      |
| @DOMAIN      | Domain name of a computer            |
| @DOW         | Short name of day of week for date   |
| @DOWF        | Full name of day of week for date    |
| @DOWI        | Day of week number for date          |
| @DOY         | Day of year for date                 |
| @DRIVETYPE   | Type of a drive                      |
| @DRIVETYPEEX | Type of a drive                      |

|              |                                                              |
|--------------|--------------------------------------------------------------|
| @ENUMSOURCES | Identify server names on a network                           |
| @ENUMSHARES  | Identify sharenames on a server                              |
| @ERRTEXT     | Windows error description                                    |
| @EVAL        | Arithmetic calculations                                      |
| @EXEC        | Execute a command and return its exit code                   |
| @EXECARRAY   | Execute a command and store the results in an array variable |
| @EXECSTR     | Execute a command and return the first output line           |
| @EXETYPE     | Application type                                             |
| @EXPAND      | All names that match filename                                |
| @EXT         | File extension                                               |

|             |                                   |
|-------------|-----------------------------------|
| @FIELD      | Extract a field from a string     |
| @FIELDS     | Count fields in a string          |
| @FILEAGE    | File age (date and time)          |
| @FILEARRAY  | Read a file into an array         |
| @FILECLOSE  | Close a file handle               |
| @FILEDATE   | File date                         |
| @FILEHANDLE | Returns the filename for a handle |
| @FILENAME   | File name and extension           |
| @FILEOPEN   | Open a file handle                |
| @FILEREAD   | Read next line from a file        |

|             |                                                              |
|-------------|--------------------------------------------------------------|
| @FILEREADB  | Read bytes from a file                                       |
| @FILES      | Number of files matching a wildcard                          |
| @FILESEEK   | Move a file handle pointer to specified file position        |
| @FILESEEKL  | Move a file handle pointer to a specified line               |
| @FILESIZE   | Total size of files matching a wildcard                      |
| @FILETIME   | File time                                                    |
| @FILEWRITE  | Write next line to a file                                    |
| @FILEWRITEB | Write data to a file                                         |
| @FILTER     | Removes non-matching characters from a string                |
| @FINDCLOSE  | Closes the search handle.                                    |
| @FINDFIRST  | Find first matching file                                     |
| @FINDNEXT   | Find next matching file                                      |
| @FLOOR      | Largest integer not larger than a number                     |
| @FOLDERS    | Number of folders                                            |
| @FORMAT     | Formats data string according to format string               |
| @FONT       | Return console font information                              |
| @FORMATN    | Format a numeric value                                       |
| @FORMATNC   | Format a numeric value and insert the thousands separator(s) |
| @FSTYPE     | File system type (FAT, NTFS, CDFS, etc.)                     |
| @FTYPE      | Open command string for file type                            |
| @FULL       | Full file name with path                                     |
| @FUNCTION   | Definition of a function                                     |

|              |                                              |
|--------------|----------------------------------------------|
| @GETDATE     | Select a date from a calendar                |
| @GETDATETIME | Select a date and/or time from a date picker |
| @GETDIR      | Prompt for a directory name.                 |
| @GETFILE     | Prompt for a path and file name.             |
| @GETFOLDER   | Folder name from tree view.                  |
| @GROUP       | User is member of group: 1, otherwise 0      |

|             |                                         |
|-------------|-----------------------------------------|
| @HEXDECODE  | Decode hexadecimal file or string       |
| @HEXENCODE  | Encode file or string as hexadecimal    |
| @HISTORY    | A line or word from the command history |
| @HTMLDECODE | Decode an HTML escaped string           |
| @HTMLENCODE | Encode a string for HTML                |

|             |                                          |
|-------------|------------------------------------------|
| @IDOW       | Short local name of day_of_week for date |
| @IDOWF      | Full local name of day of week for date  |
| @IF         | Evaluates a conditional expression       |
| @INC        | Increment a numeric value by 1           |
| @INDEX      | Offset of string2 within string1         |
| @INIREAD    | Return an entry from an .INI file        |
| @INIVWRITE  | Write an entry in an .INI file           |
| @INSERT     | Inserts string1 into string2             |
| @INODE      | File Inode (in hex)                      |
| @INSTR      | Extract a substring                      |
| @INT        | Integer part of a number                 |
| @IPADDRESS  | Returns the numeric IP for a host name   |
| @IPADDRESSN | Address of an adapter                    |
| @IPALIASES  | Aliases of a host name                   |
| @IPDESC     | Description of an adapter                |
| @IPDHCP     | DHCP server for an adapter               |

|                   |                                                    |
|-------------------|----------------------------------------------------|
| @IPGATEWAY        | Gateway for an adapter                             |
| @IPIPV6           | IPv6 address of an adapter                         |
| @IPNAME           | Returns the host name for a numeric IP address     |
| @IPNAMEN          | Name of an adapter                                 |
| @IPPHYSICAL       | Physical address of an adapter                     |
| @IPPORT           | Port number for a service                          |
| @IPSERVICEALIASES | Aliases for a service                              |
| @IPSUBNET         | Subnet of an adapter                               |
| @IPTYPE           | Type of an adapter                                 |
| @IPWINS           | Returns 1 if the adapter is using WINS             |
| @IPWINSSERVER     | Primary WINS server for an adapter                 |
| @IPZONEID         | IPv6 Zone ID for an adapter                        |
| @ISALNUM          | Test for alphanumeric characters                   |
| @ISALPHA          | Test for alphabetic characters                     |
| @ISASCII          | Test for ASCII characters                          |
| @ISCNTRL          | Test for control characters                        |
| @ISDIGIT          | Test for decimal digits                            |
| @ISFLOAT          | Returns 1 if the string is a floating point number |
| @ISLOWER          | Returns 1 if the string is only lower-case letters |
| @ISODOWI          | ISO 8601 numeric day of week                       |
| @ISOYEAR          | ISO 8601 numeric week of year                      |
| @ISOWYEAR         | ISO 8601 numeric week date year                    |
| @ISPRIME          | Test for prime number                              |
| @ISPRINT          | Test for printable characters                      |
| @ISPROC           | Returns 1 if the process is active; otherwise 0    |
| @ISPUNCT          | Test for punctuation characters                    |
| @ISUPPER          | Returns 1 if the string is only upper-case letters |
| @ISSPACE          | Test for white space characters                    |
| @ISXDIGIT         | Test for hexadecimal digits                        |

|           |                                      |
|-----------|--------------------------------------|
| @JUNCTION | Directory referenced by the junction |
|-----------|--------------------------------------|

|        |                                        |
|--------|----------------------------------------|
| @LABEL | Volume label                           |
| @LCS   | Longest common sequence in two strings |
| @LEFT  | Left end of string                     |
| @LEN   | Length of a string                     |
| @LFN   | Long name for a short filename         |
| @LINE  | Specified line from a file             |
| @LINES | Count of lines in a file               |
| @LINKS | Number of NTFS hard links for the file |
| @LOWER | Convert string to lower case           |
| @LTRIM | Left trim specified characters.        |
| @ LUA  | Execute a Lua expression               |

|             |                                   |
|-------------|-----------------------------------|
| @MACADDRESS | MAC address of network interface  |
| @MAKEAGE    | Convert date and time to age      |
| @MAKEDATE   | Convert number of days to date    |
| @MAKETIME   | Convert number of seconds to time |
| @MAX        | Largest integer in the list       |
| @MD5        | MD5 hash of a string or file      |
| @MIN        | Smallest integer in the list      |
| @MONTH      | Month for date                    |

|              |                                                       |
|--------------|-------------------------------------------------------|
| @MX          | Email server for the specified user address           |
| @NAME        | File name without path or extension                   |
| @NUMERIC     | Test if a string is numeric                           |
| @OPTION      | Current configuration option value                    |
| @OWNER       | Return file owner                                     |
| @PATH        | File path without name                                |
| @PERL        | Evaluate a Perl expression                            |
| @PID         | PID for the specified process name                    |
| @PIDCOMMAND  | Return startup command line for the specified process |
| @PING        | Response time from a host                             |
| @PLUGINVER   | Plugin version number (major.minor.build)             |
| @PPID        | PID for the parent of the specified process name      |
| @PRIME       | Generate a prime number                               |
| @PRIORITY    | Priority class for the specified process              |
| @PROCESSTIME | Process times (start, end, kernel mode, user mode)    |
| @PYTHON      | Evaluate a Python expression                          |
| @QUOTE       | Double quote the argument if necessary                |
| @RANDOM      | Generate a random integer                             |
| @READSCR     | Read characters from the screen                       |
| @READY       | Drive ready: 1, otherwise 0                           |
| @REGBREAD    | Read a registry value to a binary buffer              |
| @REGBWRITE   | Write a registry value from a binary buffer           |
| @REGCOPYKEY  | Recursively copy a registry key to a new location     |
| @REGCREATE   | Create registry subkey                                |
| @REGDELKEY   | Delete a registry key and its subkeys                 |
| @REGEX       | Match a regular expression                            |
| @REGEXINDEX  | Return the offset of a regular expression match       |
| @REGEXIST    | Test if a registry key exists                         |
| @REGEXSUB    | Return nth matching regular expression group          |
| @REGQUERY    | Read value from registry                              |
| @REGSET      | Write value to registry                               |
| @REGSETEVNT  | Write value to registry and broadcast change.         |
| @REGTYPE     | Return type of registry variable                      |
| @REMOTE      | Remote (network) drive: 1, otherwise 0                |
| @REMOVABLE   | Removable drive: 1, otherwise 0                       |
| @REPEAT      | Repeat a character                                    |
| @REPLACE     | Replace string1 with string2 in text                  |
| @REREPLACE   | Regular expression back reference substitution        |
| @REVERSE     | Reverse a string                                      |
| @REXX        | Value of executing an expression by REXX              |
| @RIGHT       | Right end of string.                                  |
| @RTRIM       | Removes specified trailing characters.                |
| @RUBY        | Evaluate a Ruby expression                            |
| @SCRIPT      | Evaluate expression in an active scripting engine.    |

|                  |                                                 |
|------------------|-------------------------------------------------|
| @SEARCH          | Path search                                     |
| @SELECT          | Menu selection                                  |
| @SERIAL          | Serial number of a disk                         |
| @SERIALPORTCLOSE | Close the serial port                           |
| @SERIALPORTFLUSH | Flush the serial port buffer                    |
| @SERIALPORTOPEN  | Open a serial port                              |
| @SERIALPORTREAD  | Read the serial port buffer                     |
| @SERIALPORTWRITE | Write a string to the serial port buffer        |
| @SERVER          | Query server information                        |
| @SERVICE         | Query service information                       |
| @SFN             | Short name for a long filename                  |
| @SHA1            | SHA1 checksum for the file                      |
| @SHA256          | SHA2-256 checksum for the file                  |
| @SHA384          | SHA2-384 checksum for the file                  |
| @SHA512          | SHA2-512 checksum for the file                  |
| @SHFOLDER        | Get Windows folder locations                    |
| @SIMILAR         | Compare two strings for similarity              |
| @SMCLOSE         | Close a shared memory handle                    |
| @SMOPEN          | Return a handle to shared memory                |
| @SMPEEK          | Read a value from shared memory                 |
| @SPOKE           | Write a value to shared memory                  |
| @SMREAD          | Read a string from shared memory                |
| @SMWRITE         | Write a string to shared memory                 |
| @SNAPSHOT        | Save a window or desktop as a BMP               |
| @STRIP           | Strips all characters in char from string       |
| @SUBST           | Substitute a string within another string       |
| @SUBSTR          | Extract a substring                             |
| @SUMMARY         | Query or set the NTFS SummaryInformation stream |
| @SYMLINK         | Target of a symbolic link                       |
| @SYSTEMTIME      | System idle, kernel, or user time               |

|              |                                                                   |
|--------------|-------------------------------------------------------------------|
| @TALNUM      | Number of alphanumeric characters in a string                     |
| @TALPHA      | Number of alphabetic characters in a string                       |
| @TARCFILE    | Compressed name of a file in a tar archive                        |
| @TARCOUNT    | Number of files in a tar archive                                  |
| @TARDFILE    | Uncompressed name of a file in a tar archive                      |
| @TARFILEDATE | Date and time of a file in a tar archive                          |
| @TARFILESIZE | Size of a file in a tar archive                                   |
| @TASCII      | Number of 7-bit ASCII characters in a string                      |
| @TCL         | Evaluate a Tcl expression                                         |
| @TCNTRL      | Number of ASCII control characters in a string                    |
| @TDIGIT      | Number of digits (0-9) in a string                                |
| @TIME        | Convert a time of day to number of seconds                        |
| @TIMER       | Get split time from timer.                                        |
| @TK          | Evaluate a Tk (Tcl) script or expression                          |
| @TLOWER      | Number of lower case characters in a string                       |
| @TPRINT      | Number of printable characters in a string                        |
| @TPUNC       | Number of punctuation characters in a string                      |
| @TRIM        | Remove leading & trailing blanks from a string                    |
| @TRIMALL     | Remove leading, trailing, and extra internal blanks from a string |
| @TRUENAME    | Find true name of a file                                          |
| @TRUNCATE    | Truncate file at current position                                 |
| @TSPACE      | Number of white space characters in a string                      |

|                                |                                                                             |
|--------------------------------|-----------------------------------------------------------------------------|
| <a href="#">@TUPPER</a>        | Number of upper case characters in a string                                 |
| <a href="#">@TXDIGIT</a>       | Number of hexadecimal digits in a string                                    |
| <a href="#">@UNC</a>           | UNC name of a file                                                          |
| <a href="#">@UNICODE</a>       | Numeric UNICODE value for a character                                       |
| <a href="#">@UNIQUE</a>        | Create file with unique name                                                |
| <a href="#">@UNQUOTE</a>       | Remove double quotes from a filename                                        |
| <a href="#">@UNQUOTES</a>      | Remove leading and trailing double quotes                                   |
| <a href="#">@UPPER</a>         | Convert string to upper case                                                |
| <a href="#">@URLDECODE</a>     | Decode an URL string                                                        |
| <a href="#">@URLENCODE</a>     | Encode an URL string                                                        |
| <a href="#">@UTF8DECODE</a>    | Decode a UTF8 file or string                                                |
| <a href="#">@UTF8ENCODE</a>    | Encode a file or string as UTF8                                             |
| <a href="#">@UUDECODE</a>      | Decode a UU Encoded file                                                    |
| <a href="#">@UUENCODE</a>      | Encode a file as UU Encoded                                                 |
| <a href="#">@UUID</a>          | Returns a UUID (GUID)                                                       |
| <a href="#">@VERINFO</a>       | Executable file version information                                         |
| <a href="#">@VERSION</a>       | Returns a versioned replacement for a filename                              |
| <a href="#">@WATTRIB</a>       | Test or return file attributes                                              |
| <a href="#">@WILD</a>          | Compares strings using wildcards                                            |
| <a href="#">@WINAPI</a>        | Call a Windows API function                                                 |
| <a href="#">@WINCLASS</a>      | Title of first window with class name                                       |
| <a href="#">@WINCLIENTSIZE</a> | Client window size                                                          |
| <a href="#">@WINEXENAME</a>    | Executable name for window                                                  |
| <a href="#">@WININFO</a>       | Current system information                                                  |
| <a href="#">@WINMEMORY</a>     | Windows memory information                                                  |
| <a href="#">@WINMETRICS</a>    | Windows system metrics                                                      |
| <a href="#">@WINPID</a>        | Process ID for a window                                                     |
| <a href="#">@WINPOS</a>        | Window position                                                             |
| <a href="#">@WINSIZE</a>       | Window size                                                                 |
| <a href="#">@WINSTATE</a>      | Current state of window                                                     |
| <a href="#">@WINSYSTEM</a>     | Set/get windows system parameters                                           |
| <a href="#">@WMI</a>           | Query WMI                                                                   |
| <a href="#">@WORD</a>          | Extract a word from a string                                                |
| <a href="#">@WORDS</a>         | Count words in a string                                                     |
| <a href="#">@WORKGROUP</a>     | Workgroup name of a computer                                                |
| <a href="#">@XMLCLOSE</a>      | Close an XML file previously opened by @XMLOPEN                             |
| <a href="#">@XMLNODES</a>      | Return the number of nodes (children) for the specified path in an XML file |
| <a href="#">@XMLOPEN</a>       | Open an XML file for use by @XMLPATH and/or @XMLNODES                       |
| <a href="#">@XMLPATH</a>       | Return text of XML element                                                  |
| <a href="#">@YDECODE</a>       | Decode a Y Encoded file or string                                           |
| <a href="#">@YEAR</a>          | Year for date                                                               |
| <a href="#">@YENCODE</a>       | Encode a file or string as Y Encoded                                        |
| <a href="#">@ZIPCFILE</a>      | The compressed name of a file in a zip archive                              |
| <a href="#">@ZIPCOMMENT</a>    | The comment for a zip archive                                               |

|                                 |                                                      |
|---------------------------------|------------------------------------------------------|
| <a href="#">@ZIPCOUNT</a>       | The number of files in a zip archive                 |
| <a href="#">@ZIPFILE</a>        | The decompressed name of a file in a zip archive     |
| <a href="#">@ZIPFILECOMMENT</a> | The comment (description) of a file in a zip archive |
| <a href="#">@ZIPFILECRC</a>     | The CRC of a file in a zip archive                   |
| <a href="#">@ZIPFILEDATE</a>    | The date and time of a file in a zip archive         |
| <a href="#">@ZIPCFILESIZE</a>   | The compressed size of a file in a zip archive       |
| <a href="#">@ZIPDFILESIZE</a>   | The decompressed size of a file in a zip archive     |

#### 4.4.5.2 Functions by Category

See also [Functions by Name](#).

This list gives a one-line description of all built-in [Variable Functions](#), and a cross reference which selects a separate help topic on that function where you will find the detailed syntax and description. You can also obtain help on any function with a **HELP @functionname** command at the prompt or by pressing [Ctrl-F1](#) when the cursor is on the function name. See the [HELP](#) command for details

- [Binary buffers](#)
- [Input dialog boxes](#)
- [Compression](#)
- [Network properties](#)
- [Dates and times](#)
- [Numbers and arithmetic](#)
- [Drives and devices](#)
- [Strings and characters](#)
- [File content](#)
- [System status](#)
- [File names](#)
- [Utility](#)
- [File properties](#)

Note: many functions have functionality that covers several categories.

#### System status

|                              |                                                    |
|------------------------------|----------------------------------------------------|
| <a href="#">@ASSOC</a>       | File association for the extension                 |
| <a href="#">@CLIP</a>        | Specified line from clipboard                      |
| <a href="#">@CLIPW</a>       | Write string to the clipboard                      |
| <a href="#">@CONSOLE</a>     | Identify console sessions                          |
| <a href="#">@CONSOLEB</a>    | Create or restore console screen buffers           |
| <a href="#">@ERRTEXT</a>     | Windows error description                          |
| <a href="#">@FTYPE</a>       | Open command string for the file type              |
| <a href="#">@ISPROC</a>      | Returns 1 if a process is active; otherwise 0      |
| <a href="#">@PID</a>         | Process ID for the process name                    |
| <a href="#">@PIDCOMMAND</a>  | Startup command line for a process                 |
| <a href="#">@PPID</a>        | Process ID of the parent the process name          |
| <a href="#">@PRIORITY</a>    | Priority class for a process                       |
| <a href="#">@PROCESSTIME</a> | Process times (start, end, kernel mode, user mode) |
| <a href="#">@READSCR</a>     | Read characters from the screen                    |
| <a href="#">@REGBREAD</a>    | Read registry value into a binary buffer           |
| <a href="#">@REGBWRITE</a>   | Write registry value from a binary buffer          |
| <a href="#">@REGCOPYKEY</a>  | Recursively copy a registry key to a new location  |
| <a href="#">@REGCREATE</a>   | Create registry subkey                             |
| <a href="#">@REGDELKEY</a>   | Delete a registry key and its subkeys              |
| <a href="#">@REGEXIST</a>    | Test if a registry key exists                      |
| <a href="#">@REGQUERY</a>    | Read value from registry                           |
| <a href="#">@REGSET</a>      | Write value to registry                            |
| <a href="#">@REGSETENV</a>   | Write value to registry and broadcast change.      |
| <a href="#">@REGTYPE</a>     | Type of registry variable                          |

|                  |                                          |
|------------------|------------------------------------------|
| @SERIALPORTCLOSE | Close the serial port                    |
| @SERIALPORTFLUSH | Flush the serial port buffer             |
| @SERIALPORTOPEN  | Open a serial port                       |
| @SERIALPORTREAD  | Read the serial port buffer              |
| @SERIALPORTWRITE | Write a string to the serial port buffer |
| @SYSTEMTIME      | System times (idle, kernel, and user)    |
| @WINCLASS        | Title of first window with classname     |
| @WINCLIENTSIZE   | Client window size                       |
| @WINEXENAME      | Executable name for window               |
| @WININFO         | Current system information               |
| @WINMEMORY       | Windows memory information               |
| @WINMETRICS      | Windows system metrics                   |
| @WINPID          | Process ID for window                    |
| @WINPOS          | Window position                          |
| @WINSIZE         | Window size                              |
| @WINSTATE        | Current state of window                  |
| @WINSYSTEM       | Set/get windows system parameters        |
| @WINTITLE        | Window title of PID                      |

### Directories, drives and devices

|              |                                                                       |
|--------------|-----------------------------------------------------------------------|
| @CDROM       | CD-ROM drive: 1, otherwise 0                                          |
| @CWD         | Current Working Directory of specified drive                          |
| @CWDS        | Current Working Directory of specified drive, with trailing backslash |
| @DEVICE      | Character device: 1, otherwise 0                                      |
| @DISKFREE    | Free disk space                                                       |
| @DISKTOTAL   | Total disk space                                                      |
| @DISKUSED    | Used disk space                                                       |
| @DRIVETYPE   | Type of drive (hard drive, CD-ROM, etc.)                              |
| @DRIVETYPEEX | Type of drive (hard drive, CD-ROM, etc.)                              |
| @FSTYPE      | File system type (FAT, NTFS, CDFS, etc.)                              |
| @JUNCTION    | Directory referenced by the junction                                  |
| @LABEL       | Volume label                                                          |
| @READY       | Drive ready: 1, otherwise 0                                           |
| @REMOTE      | Remote (network) drive: 1, otherwise 0                                |
| @REMOVABLE   | Removable drive: 1, otherwise 0                                       |
| @SERIAL      | Serial number of a disk                                               |
| @SHFOLDER    | Windows folder locations                                              |
| @SYMLINK     | Target of a symbolic link                                             |

### File content

|            |                                   |
|------------|-----------------------------------|
| @B64DECODE | Decode a Base64 file or string    |
| @B64ENCODE | Encode a file or string as Base64 |
| @COMPARE   | Compare two files                 |
| @CRC32     | File CRC                          |
| @FILEARRAY | Read a file into an array         |
| @FILECLOSE | Close a file handle               |
| @FILEOPEN  | Open a file handle                |
| @FILEREAD  | Read next line from a file        |
| @FILEREADB | Read bytes from a file            |
| @FILESEEK  | Move a file handle pointer        |

|             |                                                |
|-------------|------------------------------------------------|
| @FILESEEKL  | Move a file handle pointer to a specified line |
| @FILEWRITE  | Write next line to a file                      |
| @FILEWRITEB | Write data to a file handle                    |
| @HEXDECODE  | Decode a hexadecimal file or string            |
| @HEXENCODE  | Encode a file or string as hexadecimal         |
| @INIREAD    | Return an entry from an .INI file              |
| @INIWRITE   | Write an entry in an .INI file                 |
| @INODE      | Inode value for a file                         |
| @LINE       | Specified line from a file                     |
| @LINES      | Count lines in a file                          |
| @LINKS      | Number of NTFS hard links for a file           |
| @MD5        | MD5 hash of a string or file                   |
| @SHA1       | SHA1 checksum for a file                       |
| @SHA256     | SHA2-256 checksum for a file                   |
| @SHA384     | SHA2-384 checksum for a file                   |
| @SHA512     | SHA2-512 checksum for a file                   |
| @SUMMARY    | NTFS Summary\Information stream for a file     |
| @TRUNCATE   | Truncate file at current position              |
| @UTF8DECODE | Decode a UTF8 file or string                   |
| @UTF8ENCODE | Encode a file or string as UTF8                |
| @UUDECODE   | Decode a UU Encoded file                       |
| @UUENCODE   | Encode a file as UU Encoded                    |
| @VERINFO    | Executable file version information            |
| @YDECODE    | Decode a Y Encoded file or string              |
| @YENCODE    | Encode a file or string as Y Encoded           |

### File names

|             |                                           |
|-------------|-------------------------------------------|
| @ALTPATH    | Short name for the file.                  |
| @EXPAND     | All names that match filename             |
| @EXT        | File extension                            |
| @FILEHANDLE | Filename for a handle                     |
| @FILENAME   | File name and extension                   |
| @FULL       | Full file name with path                  |
| @LFN        | Long name for a short filename            |
| @NAME       | File name without path or extension       |
| @PATH       | File path without name                    |
| @QUOTE      | Double quote a filename                   |
| @SFN        | Short name for a long filename            |
| @SEARCH     | Path search                               |
| @TRUENAME   | True name of a file                       |
| @UNC        | UNC name of a file                        |
| @UNIQUE     | Create file with unique name              |
| @UNQUOTE    | Remove double quotes from a filename      |
| @UNQUOTES   | Remove leading and trailing double quotes |
| @VERSION    | Returns a versioned filename              |

### File properties

|           |                                |
|-----------|--------------------------------|
| @ATTRIB   | Test or return file attributes |
| @DESCRIPT | File description               |
| @EXETYPE  | Application type               |

|            |                                           |
|------------|-------------------------------------------|
| @FILEAGE   | File age (date and time)                  |
| @FILEDATE  | File date                                 |
| @FILES     | Number of files matching a wildcard       |
| @FILESIZE  | Total size of files matching a wildcard   |
| @FILETIME  | File time                                 |
| @FINDCLOSE | Closes the search handle.                 |
| @FINDFIRST | Find first matching file                  |
| @FINDNEXT  | Find next matching file                   |
| @INODE     | Inode value for a file                    |
| @LINKS     | Number of NTFS hard links for a file      |
| @OWNER     | File owner                                |
| @SEARCH    | Path search                               |
| @SUMMARY   | NTFS SummaryInformation stream for a file |
| @TRUENAME  | True name for a file                      |
| @UNIQUE    | Create file with unique name              |
| @VERINFO   | Executable file version information       |
| @WATTRIB   | Test or return file attributes            |

### Strings and characters

|             |                                                              |
|-------------|--------------------------------------------------------------|
| @ASCII      | List of ASCII-s for characters in string                     |
| @CAPS       | Capitalize first character of each word                      |
| @CHAR       | Character string, given a set of ASCII-s                     |
| @COUNT      | Counts occurrences of a character in a string                |
| @EXECARRAY  | Execute a command and store the results in an array variable |
| @EXECSTR    | Execute a command and return the output line                 |
| @FIELD      | Extract a field from a string                                |
| @FIELDS     | Count fields in a string                                     |
| @FILTER     | Removes non-matching characters from a string                |
| @FORMAT     | Formats data string according to format string               |
| @HTMLDECODE | Decode an HTML string                                        |
| @HTMLENCODE | Encode a string for HTML                                     |
| @INDEX      | Offset of string2 within string1                             |
| @INSERT     | Insert string1 into string2                                  |
| @INSTR      | Extract a substring                                          |
| @ISALNUM    | Test for alphanumeric characters                             |
| @ISALPHA    | Test for alphabetic characters                               |
| @ISASCII    | Test for ASCII characters                                    |
| @ISCNTRL    | Test for control characters                                  |
| @ISDIGIT    | Test for decimal digits                                      |
| @ISFLOAT    | Returns 1 if the string is a floating point number           |
| @ISLOWER    | Returns 1 if the string is all lower case                    |
| @ISPRINT    | Test for printable characters                                |
| @ISPUNCT    | Test for punctuation characters                              |
| @ISSPACE    | Test for white space characters                              |
| @ISUPPER    | Returns 1 if the string is all upper case                    |
| @ISXDIGIT   | Test for hexadecimal digits                                  |
| @LCS        | Longest common sequence in two strings                       |
| @LEFT       | Left end of string                                           |
| @LEN        | Length of a string                                           |
| @LOWER      | Convert string to lower case                                 |
| @LTRIM      | Trims specified leading characters.                          |
| @MD5        | MD5 hash of a string or file                                 |

|             |                                                                     |
|-------------|---------------------------------------------------------------------|
| @MX         | Email server for a user address                                     |
| @QUOTE      | Double quote a string                                               |
| @REGEX      | Return a Regular Expression test                                    |
| @REGEXINDEX | Return the offset of a regular expression match                     |
| @REGEXSUB   | Return the nth matching group of a regular expression test          |
| @REPEAT     | Repeat a character                                                  |
| @REPLACE    | Replace string1 with string2 in text                                |
| @REREPLACE  | Regular expression back reference substitution                      |
| @REVERSE    | Reverse a string                                                    |
| @RIGHT      | Right end of string                                                 |
| @RTRIM      | Trims specified trailing characters.                                |
| @SIMILAR    | Test similarity between two strings                                 |
| @STRIP      | Strips all characters in char from string                           |
| @SUBST      | Substitute a string within another string                           |
| @SUBSTR     | Older version of @INSTR to extract a substring                      |
| @TALNUM     | Number of alphanumeric characters in a string                       |
| @TALPHA     | Number of alphabetic characters in a string                         |
| @TASCII     | Number of 7-bit ASCII characters in a string                        |
| @TCNTRL     | Number of ASCII control characters in a string                      |
| @TDIGIT     | Number of digits (0-9) in a string                                  |
| @TLOWER     | Number of lower case characters in a string                         |
| @TPRINT     | Number of printable characters in a string                          |
| @TPUNCT     | Number of punctuation characters in a string                        |
| @TRIM       | Remove leading and trailing blanks from a string                    |
| @TRIMALL    | Removing leading, trailing, and extra internal blanks from a string |
| @TSPACE     | Number of white space characters in a string                        |
| @TUPPER     | Number of upper case characters in a string                         |
| @TXDIGIT    | Number of hexadecimal digits in a string                            |
| @UNICODE    | List of UNICODEs for characters in string                           |
| @UNQUOTE    | Remove double quotes from a string                                  |
| @UNQUOTES   | Remove leading and trailing double quotes                           |
| @UPPER      | Convert string to upper case                                        |
| @URLDECODE  | Decode an URL string                                                |
| @URLENCODE  | Encode an URL string                                                |
| @WILD       | Compares strings using wildcards                                    |
| @WORD       | Extract a word from a string                                        |
| @WORDS      | Count words in a string                                             |

### Binary buffers and shared memory

|           |                                      |
|-----------|--------------------------------------|
| @BALLOC   | Allocate a binary buffer             |
| @BFREE    | Free a binary buffer                 |
| @BPEEK    | Read a value from a binary buffer    |
| @BPEEKSTR | Read a string from a binary buffer   |
| @BPOKE    | Write a value to a binary buffer     |
| @BPOKESTR | Write a string to a binary buffer    |
| @BREAD    | Read from a file to a binary buffer  |
| @BWRITE   | Write from a binary buffer to a file |
| @SMCLOSE  | Close a handle to shared memory      |
| @SMOPEN   | Open a handle to shared memory       |
| @SMPEEK   | Read a value from shared memory      |

|                          |                                  |
|--------------------------|----------------------------------|
| <a href="#">@SMPOKE</a>  | Write a value to shared memory   |
| <a href="#">@SMREAD</a>  | Read a string from shared memory |
| <a href="#">@SMWRITE</a> | Write a string to shared memory  |

## Numbers and arithmetic

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| <a href="#">@ABS</a>      | Absolute value of n                                        |
| <a href="#">@AVERAGE</a>  | Average of a list                                          |
| <a href="#">@CEILING</a>  | Smallest integer not less than n                           |
| <a href="#">@COMMA</a>    | Insert commas (thousands separators) into a numeric string |
| <a href="#">@CONVERT</a>  | Convert value from input base to output base               |
| <a href="#">@DEC</a>      | Decrement a numeric value by 1                             |
| <a href="#">@DECIMAL</a>  | Decimal fraction portion of a number                       |
| <a href="#">@DIGITS</a>   | Tests if string is all digits                              |
| <a href="#">@EVAL</a>     | Arithmetic calculations                                    |
| <a href="#">@FORMATN</a>  | Format a numeric value                                     |
| <a href="#">@FORMATNC</a> | Format a numeric value and insert thousands separators     |
| <a href="#">@FLOOR</a>    | Largest integer not larger than n                          |
| <a href="#">@INC</a>      | Increment a numeric value by 1                             |
| <a href="#">@INT</a>      | Integer part of a number                                   |
| <a href="#">@ISPRIME</a>  | Test if a number is a prime                                |
| <a href="#">@MAX</a>      | Largest integer in the list                                |
| <a href="#">@MIN</a>      | Smallest integer in the list                               |
| <a href="#">@NUMERIC</a>  | Test if a string is numeric                                |
| <a href="#">@PRIME</a>    | Generate a prime number                                    |
| <a href="#">@RANDOM</a>   | Generate a random integer                                  |

## Dates and times

|                              |                                              |
|------------------------------|----------------------------------------------|
| <a href="#">@AGEDATE</a>     | Converts an age into date and time           |
| <a href="#">@DAY</a>         | Day of month for date                        |
| <a href="#">@DATE</a>        | Convert date to number of days               |
| <a href="#">@DATECONV</a>    | Convert date formats                         |
| <a href="#">@DOW</a>         | Short name of day of week for date           |
| <a href="#">@DOWF</a>        | Full name of day of week                     |
| <a href="#">@DOWI</a>        | Day of week as integer                       |
| <a href="#">@DOY</a>         | Day of year for date                         |
| <a href="#">@GETDATE</a>     | Select a date from a calendar                |
| <a href="#">@GETDATETIME</a> | Select a date and/or time from a date picker |
| <a href="#">@IDOW</a>        | Short localized name of day of week for date |
| <a href="#">@IDOWF</a>       | Full localized name of day of week for date  |
| <a href="#">@ISODOWI</a>     | ISO 8601 numeric day of week                 |
| <a href="#">@ISOWEEK</a>     | ISO 8601 numeric week of year                |
| <a href="#">@ISOWYEAR</a>    | ISO 8601 numeric week date year              |
| <a href="#">@MAKEAGE</a>     | Convert date and time to age                 |
| <a href="#">@MAKEDATE</a>    | Convert number of days to date               |
| <a href="#">@MAKETIME</a>    | Convert number of seconds to time            |
| <a href="#">@MONTH</a>       | Month in specified date                      |
| <a href="#">@TIME</a>        | Convert time to number of seconds            |
| <a href="#">@YEAR</a>        | Year for date                                |

### Input dialog boxes

|                            |                                  |
|----------------------------|----------------------------------|
| <a href="#">@GETDIR</a>    | Prompt for a directory name.     |
| <a href="#">@GETFILE</a>   | Prompt for a path and file name. |
| <a href="#">@GETFOLDER</a> | Folder name from tree view.      |
| <a href="#">@SELECT</a>    | Menu selection                   |

### Network properties

|                                   |                                         |
|-----------------------------------|-----------------------------------------|
| <a href="#">@AFSCELL</a>          | OpenAFS cell name for a path            |
| <a href="#">@AFSMOUNT</a>         | OpenAFS mount point for a path          |
| <a href="#">@AFSPATH</a>          | Path is in OpenAFS: 1, otherwise 0      |
| <a href="#">@AFSSYMLINK</a>       | OpenAFS symbolic link for a path        |
| <a href="#">@AFSVOLD</a>          | OpenAFS volume ID for a path            |
| <a href="#">@AFSVOLNAME</a>       | OpenAFS volume name for a path          |
| <a href="#">@DOMAIN</a>           | Domain name of a computer               |
| <a href="#">@ENUMSERVERS</a>      | Identify server names on a network      |
| <a href="#">@ENUMSHARES</a>       | Identify sharenames on a server         |
| <a href="#">@GROUP</a>            | User is member of group: 1, otherwise 0 |
| <a href="#">@IPADDRESS</a>        | The numeric IP for a host name          |
| <a href="#">@IPADDRESSN</a>       | IP address of an adapter                |
| <a href="#">@IPALIASES</a>        | Aliases of a host                       |
| <a href="#">@IPDESC</a>           | Description of an adapter               |
| <a href="#">@IPDHCP</a>           | DHCP server of an adapter               |
| <a href="#">@IPGATEWAY</a>        | Gateway of an adapter                   |
| <a href="#">@IPIPV6N</a>          | IPv6 address of an adapter              |
| <a href="#">@IPNAME</a>           | The host name for a numeric IP          |
| <a href="#">@IPNAMEN</a>          | Name of an adapter                      |
| <a href="#">@IPPHYSICAL</a>       | Physical address of an adapter          |
| <a href="#">@IPPORT</a>           | Port number for a service               |
| <a href="#">@IPSERVICEALIASES</a> | Aliases for a service                   |
| <a href="#">@IPSUBNET</a>         | Subnet mask of an adapter               |
| <a href="#">@IPTYPE</a>           | Type of an adapter                      |
| <a href="#">@IPWINS</a>           | Returns 1 if an adapter is using WINS   |
| <a href="#">@IPWINSSERVER</a>     | Primary WINS server for an adapter      |
| <a href="#">@IPZONEID</a>         | IPv6 Zone ID of an adapter              |
| <a href="#">@MACADDRESS</a>       | MAC address of network interface        |
| <a href="#">@PING</a>             | Response time from a host               |
| <a href="#">@SERVER</a>           | Query server information                |
| <a href="#">@WORKGROUP</a>        | Workgroup name of a computer            |

### Utility

|                            |                                |
|----------------------------|--------------------------------|
| <a href="#">@ALIAS</a>     | Value of an alias              |
| <a href="#">@ARRAYINFO</a> | Array variable information     |
| <a href="#">@CAPI</a>      | Call a cdecl function in a DLL |
| <a href="#">@CLIP</a>      | Specified line from clipboard  |
| <a href="#">@CLIPW</a>     | Write string to the clipboard  |
| <a href="#">@COLOR</a>     | RGB value of a color           |
| <a href="#">@DEBUG</a>     | Write a string to the debugger |

|            |                                                                             |
|------------|-----------------------------------------------------------------------------|
| @DIRSTACK  | Display directory stack entry                                               |
| @ERRTEXT   | Windows error description                                                   |
| @EXEC      | Execute a command, returns its exit code                                    |
| @EXECSTR   | Execute a command, returns its first output line                            |
| @FONT      | Console font information                                                    |
| @FUNCTION  | Definition of a function                                                    |
| @HISTORY   | A line or word from the command history                                     |
| @IF        | Value dependent on a conditional expression                                 |
| @LUA       | Execute a Lua expression                                                    |
| @OPTION    | Current configuration option value                                          |
| @PERL      | Evaluate a Perl expression                                                  |
| @PLUGINVER | Plugin version number                                                       |
| @READSCR   | Read characters from the screen                                             |
| @REXX      | Evaluate a REXX expression                                                  |
| @RUBY      | Evaluate a Ruby expression                                                  |
| @SCRIPT    | Evaluate expression in active scripting engine                              |
| @SELECT    | Menu selection                                                              |
| @SERVICE   | Query service information                                                   |
| @SNAPSHOT  | Save a window or the desktop to a BMP                                       |
| @TCL       | Execute a Tcl/tk command                                                    |
| @TIMER     | Get split time from timer.                                                  |
| @TK        | Execute a Tk script or expression                                           |
| @UUID      | Create a UUID / GUID                                                        |
| @WINAPI    | Call a Windows API function                                                 |
| @WMI       | Query WMI                                                                   |
| @XMLCLOSE  | Close an XML file previously opened by @XMLOPEN                             |
| @XMLNODES  | Return the number of nodes (children) for the specified path in an XML file |
| @XMLOPEN   | Open an XML file for use by @XMLPATH and/or @XMLNODES                       |
| @XMLPATH   | Return text of XML element                                                  |

### Compression and Decompression

|                 |                                                       |
|-----------------|-------------------------------------------------------|
| @TARCOUNT       | The number of files in a .tar archive                 |
| @TARCFILE       | The compressed name of a file in a .tar archive       |
| @TARDFILE       | The decompressed name of a file in a .tar archive     |
| @TARFILEDATE    | The date and time of a file in a .tar archive         |
| @TARFILESIZE    | The (uncompressed) size of a file in a .tar archive   |
| @ZIPCOUNT       | The number of files in a .zip archive                 |
| @ZIPCOMMENT     | The comment text for a .zip archive                   |
| @ZIPCFILE       | The compressed name of a file in a .zip archive       |
| @ZIPDFILE       | The decompressed name of a file in a .zip archive     |
| @ZIPFILECOMMENT | The comment (description) of a file in a .zip archive |
| @ZIPFILECRC     | The CRC of a file in a .zip archive                   |
| @ZIPFILEDATE    | The date and time of a file in a .zip archive         |
| @ZIPCFILESIZE   | The compressed size of a file in a .zip archive       |
| @ZIPDFILESIZE   | The decompressed size of a file in a .zip archive     |

#### 4.4.5.3 Date Display Formats

All functions which **return** a date accept an **optional code** to specify the desired format of the date value:

| Code             | Date Format       | Description    |
|------------------|-------------------|----------------|
| <b>0</b> or none | <b>see below</b>  | system default |
| <b>1</b>         | <b>mm/dd/yy</b>   | USA            |
| <b>2</b>         | <b>dd/mm/yy</b>   | European       |
| <b>3</b>         | <b>yy/mm/dd</b>   | Japanese       |
| <b>4</b>         | <b>yyyy-mm-dd</b> | ISO 8601       |
| <b>5</b>         | <b>yyyy-Www-d</b> | ISO 8601       |
| <b>6</b>         | <b>yyyy-ddd</b>   | ISO 8601       |

### Field Order

For codes **1...6** the field order is as shown above. For code **0** the field order will also be one of those shown above. **TCC** determines which field is reported first by Windows in a short date, and selects the order from the table above with the same first field. All other aspects of the Windows short date format are ignored.

### Field Width

Month and day are always 2 digits. Year is 2 digits for codes **1, 2** and **3**, and 4 digits for codes **4, 5**, and **6**. For code **0** the year is 4 digits if it is the first field returned, and 2 digits if it is the last one.

### Field Separator

Codes **4, 5**, and **6** (ISO 8601) uses a hyphen as the separator character. For the other formats, the default Windows date separator is returned.

#### 4.4.5.4 @ABS

**@ABS[n]** : Returns the absolute value of the number *n*.

##### Examples:

```
echo %@abs [-1]
1
```

```
echo %@abs [123]
123
```

#### 4.4.5.5 @AFSCELL

**@AFSCELL[path]** : Returns the [OpenAFS](#) cell name for the path.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.5.6 @AFSMOUNT

**@AFSMOUNT[path]** : Returns the [OpenAFS](#) mount point for the pathname.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.5.7 @AFSPATH

**@AFSPATH[*path*]** : Returns 1 if the path is in the [OpenAFS](#) file system.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.5.8 @AFSSYMLINK

**@AFSSYMLINK[*path*]** : Returns the [OpenAFS](#) symbolic link for the path.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.5.9 @AFSVOLID

**@AFSVOLID[*path*]** : Returns the [OpenAFS](#) volume ID for the path.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.5.10 @AFSVOLNAME

**@AFSVOLNAME[*path*]** : Returns the [OpenAFS](#) volume name for the path.

See <http://www.openafs.org> for more information on OpenAFS.

#### 4.4.5.11 @AGEDATE

**@AGEDATE[*n[,d]*]** : Converts an age *n* into a date and time pair, formatted according to the current country settings, or as explicitly specified by *d* (see [Date Display Formats](#)). The time is separated from the date by a comma, and is always in 24-hour format, displayed with 1 ms precision, as the examples show. The conversion does not take leap seconds into account.

**Example:**

```
for /l %n in (1,1,6) echo %n %@agedate[128551146920835000,%n]

1 05-13-08,01:11:32.083
2 13-05-08,01:11:32.083
3 08-05-13,01:11:32.083
4 2008-05-13,01:11:32.083
5 2008-W20-2,01:11:32.083
6 2008-134,01:11:32.083
```

See also: [Time Stamps](#), [@FILEAGE](#) and [@MAKEAGE](#).

#### 4.4.5.12 @ALIAS

**@ALIAS[*name*]** : Returns the contents of the specified alias as a string, or a null string if the alias doesn't exist.

When manipulating strings returned by @ALIAS you may need to disable certain special characters with [SETDOS /X](#). Otherwise, command separators, redirection characters, and other similar characters in the alias may be interpreted as part of the current command, rather than part of a simple text string.

**Examples:**

```
alias xyz=d:\path\myprog.exe -options
echo %@alias[xyz]
```

```
d:\path\myprog.exe -options
```

#### 4.4.5.13 @ALTNAMES

**@ALTNAMES[filename]** : Returns the alternate (short, "8.3" FAT-format) name for the specified file. If the **filename** is already in 8.3 format, returns the filename. If the file does not exist, returns an empty string. If **filename** contains a \, @ALTNAMES returns the SFN of the full path.

**Examples:**

```
echo %@altname["Long Name.exe"]
LONGNA~1.EXE

echo %@altname["C:\Program Files\Microsoft Office"]
C:\PROGRA~1\MICROS~4

echo %@altname["%CommonProgramFiles%"]
C:\PROGRA~1\COMMON~1
```

#### 4.4.5.14 @ARRAYINFO

**@ARRAYINFO[arrayname,option]** : Returns information about the specified array.

*arrayname* - name of the array (defined by [SETARRAY](#)) to query

*option* - the type of information:

- 0 - total number of dimensions
- 1 - number of elements in the first dimension
- 2 - number of elements in the second dimension
- 3 - number of elements in the third dimension
- 4 - number of elements in the fourth dimension
- 5 - total number of elements

@ARRAYINFO will return -1 if the array doesn't exist.

**Examples:**

```
setarray array[5,10]
echo %@arrayinfo[array,0]
2

echo %@arrayinfo[array,2]
10
```

#### 4.4.5.15 @ASCII

**@ASCII[string]** : Returns the space separated list of ASCII values of the characters in **string**. You can use the [Escape character](#) before a special character, e.g., a quote or greater than (>) sign, to include it in **string**.

**Note:** The [@UNICODE](#) function will generally return more useful values.

**Examples:**

| <b>function</b> | <b>value</b> |
|-----------------|--------------|
| %@ascii[a]      | 97           |
| %@ascii[A]      | 65           |
| %@ascii[^]      | 96           |
| %@ascii[abc]    | 97 98 99     |

**See also:** [ASCII, Key Codes and Key Names.](#)

#### 4.4.5.16 @ASSOC

**@ASSOC[.ext[,u]]** : Returns the file association for the specified extension. If the optional second argument *u* is specified, @ASSOC will look in HKCU\SOFTWARE\CLASSES.

**Example:**

```
echo %@assoc[.doc]
Word.Document.8
```

#### 4.4.5.17 @ATTRIB

**@ATTRIB[filename[,-rhsadecijlopt[,p]]]** : If you do not specify any attributes, @ATTRIB returns the attributes of the specified file in the format **RHSADECIJNOPTV**, rather than **0** or **1**. If two or more parameters are specified, @ATTRIB returns a **1** if the specified file has all the matching attribute(s); otherwise it returns a **0**. If the optional third argument ,*p* is included (partial match), then @ATTRIB will return **1** if any of the attributes match

The basic attributes for FAT volumes are:

- N** Normal (no attributes set)
- R** Read-only
- A** Archive
- H** Hidden
- S** System
- D** Directory

In addition, NTFS volumes allow display of the following extended attributes:

- E** Encrypted
- C** Compressed
- I** Not content-indexed
- J** Junction or symbolic link
- L** Junction or symbolic link
- N** Normal
- O** Offline
- P** Sparse file
- T** Temporary
- V** Virtualized

The extended attributes are displayed when @ATTRIB is invoked with a single parameter, but they cannot be specified when querying files (two or more parameters). To query files based on the extended attributes, see [@WATTRIB](#).

Attributes which are not set will be replaced with an underscore. For example, if *SECURE.DAT* has the read-only, hidden, and archive attributes set, %@ATTRIB[*SECURE.DAT*] would return *RH\_A\_\_\_\_\_*. If the file does not exist, @ATTRIB returns an empty string.

The attributes (other than **N**) can be combined (for example %@ATTRIB[*MYFILE,HS*]). For example, %@ATTRIB[*MYFILE,HS,p*] will return **1** if *MYFILE* has the hidden, system, or both attributes. Without ,**p** the function will return **1** only if *MYFILE* has both attributes.

**Filename** must be in quotes if it contains white space or special characters.

See also: [@WATTRIB](#), [Attributes Switches](#) and the [ATTRIB](#) command.

#### Examples:

```
echo %@attrib["C:\Program Files\My Program\myfile.exe",rhs,p]
echo Attributes for myfile.exe: %@attrib[myfile.exe]
```

#### 4.4.5.18 @AVERAGE

**@AVERAGE[...]** : Returns the average of a list of numbers. The average is returned as a double; you can adjust the decimal precision by running the result through [@EVAL](#) (or [@INT](#)).

#### Example:

```
echo %@average[1 3 6 8 10 13 15]
8.0
```

#### 4.4.5.19 @B64DECODE

**@B64DECODE[s,string]** : Decode a Base64 string (MIME encoding format). Returns the decoded string.  
**@B64DECODE[inputfile,outputfile]** : Decode a Base64 file (MIME encoding format). Returns 0 if the output file was successfully written.

#### Example:

```
echo %@b64decode[s,dGhpcyBpcyBhIHN0cmluZw==]
this is a string
echo %@b64decode[data.file.b64,date.file]
```

#### 4.4.5.20 @B64ENCODE

**@B64ENCODE[s,string]** : Encode a base 64 string (MIME encoding format). Returns the encoded string.  
**@B64ENCODE[inputfile,outputfile]** : Encode a base 64 file (MIME encoding format). Returns 0 if the output file was successfully written.

#### Example:

```
echo %@b64encode[s,this is a string]
dGhpcyBpcyBhIHN0cmluZw==
echo %@b64encode[data.file,date.file.b64]
```

#### 4.4.5.21 @BALLOC

**@BALLOC[size]** : Allocate a buffer for binary operations. @BALLOC returns a handle to the buffer, which must be used for the subsequent binary functions. The only limit on the number and size of binary buffers is the amount of virtual memory available.

**Example:**

```
set handle=%@balloc[128]
echo %handle
5d4f280
```

#### 4.4.5.22 @BFREE

**@BFREE[handle]** : Free a binary buffer previously allocated by [@BALLOC](#).

**Example:**

```
set handle=%@balloc[128]
echo %@bfree[%handle]
```

#### 4.4.5.23 @BPEEK

**@BPEEK[handle,offset,size]** : Read a value from a binary buffer.

*handle* - a binary handle from [@BALLOC](#)

*offset* - the byte offset in the buffer (decimal or hex)

*size* - the size of the value to read (in bytes):

- 1 - character
- 2 - short
- 4 - int
- 8 - int64

**Example:**

```
set handle=%@balloc[128]
set value=%@bpeek[%handle,0,4]
```

#### 4.4.5.24 @BPEEKSTR

**@BPEEKSTR[handle,offset,type,length]** : Read a string from a binary buffer.

*handle* - a binary handle from [@BALLOC](#)

*offset* - the byte offset in the buffer (decimal or hex)

*type* - the string type:

- a - ASCII
- u - Unicode

*length* - the maximum number of characters to read (decimal or hex)

**Example:**

```
set handle=%@balloc[128]
set value=%@bpeekstr[%handle,0,a]
```

#### 4.4.5.25 @BPOKE

**@BPOKE[*handle,offset,size,value*]** : Write a value to a binary buffer.

*handle* - a binary handle from [@BALLOC](#)

*offset* - the byte offset in the buffer (decimal or hex)

*size* - the size of the value (in bytes):

- 1 - character
- 2 - short
- 4 - int
- 8 - int64

*value* - the value to poke

@BPOKE returns 0 on success.

**Example:**

```
set handle=%@balloc[128]
set value=%@bpoke[%handle,0,4,1234]
```

#### 4.4.5.26 @BPOKESTR

**@BPOKESTR[*handle,offset,type,string*]** : Write a string to a binary buffer.

*handle* - a binary handle from [@BALLOC](#)

*offset* - the byte offset in the buffer (decimal or hex)

*type* - the type of the string to write:

- a - ASCII
- u - Unicode

*string* - the string to poke

@BPOKESTR returns 0 on success.

**Example:**

```
set handle=%@balloc[128]
set value=%@bpokestr[%handle,0,a,string value]
```

#### 4.4.5.27 @BREAD

**@BREAD**[*handle,offset,filehandle,fileoffset,length*] : Read from a file to a binary buffer.

*handle* - a binary handle from [@ALLOC](#)

*offset* - the byte offset in the buffer (decimal or hex)

*filehandle* - a file handle opened for reading (from [@FILEOPEN](#))

*fileoffset* - the read offset (from the current file position) (decimal or hex)

*length* - number of bytes to read (decimal or hex)

**@BREAD** returns the number of bytes actually read.

**Example:**

```
set fhandle=%@fileopen[filename,r]
set bhandle=%@alloc[128]
set value=%@bread[%bhandle,0,%fhandle,0,32]
```

#### 4.4.5.28 @BWRITE

**@BWRITE**[*handle,offset,filehandle,fileoffset,length*] : Write from a binary buffer to a file.

*handle* - a binary handle from [@ALLOC](#)

*offset* - the byte offset in the buffer (decimal or hex)

*filehandle* - a file handle opened for writing (from [@FILEOPEN](#))

*fileoffset* - the write offset (from the current file position) (decimal or hex)

*length* - the number of bytes to write (decimal or hex)

**@BWRITE** returns the number of bytes written

**Example:**

```
set fhandle=%@fileopen[filename,w]
set bhandle=%@alloc[128]
set value=%@bwrite[%bhandle,0,%fhandle,0,32]
```

#### 4.4.5.29 @CAPI

**@CAPI**[*module,function[,integer | PING=n | PLONG=n | PDWORD=n | NULL | BUFFER | "string"]*] : Returns the result of calling a function with a \_cdecl type in a DLL.

**module** - name of the DLL containing the function

**function** - function name (case sensitive)

**integer** - an integer value to pass to the function

**PINT** - a pointer to the integer *n*

**PLONG** - a pointer to the long integer *n*

**PDWORD** - a pointer to the DWORD *n*

**NULL** - a null pointer (0)

**BUFFER** - @CAPI will pass an address for an internal buffer for the API to return a Unicode string value.

**aBUFFER** - @CAPI will pass an address for an internal buffer for the API to return an ASCII string value.

**"string"** - text argument (this must be enclosed in double quotes). If the argument is preceded by an 'a' (i.e., a"Argument") then it is converted from Unicode to ASCII before calling the API. (Some Windows APIs only accept ASCII arguments.)

@CAPI supports a maximum of 8 arguments. The return value is either a string value returned by the API (if BUFFER or aBUFFER is specified), or the integer value returned by the API. The function must be defined as \_cdecl. If @CAPI can't find the specified function, it will append a "W" (for the Unicode version) to the function name and try again.

See also [@WINAPI](#).

#### 4.4.5.30 @CAPS

**@CAPS[["xxx"],text]** : Capitalizes the first letter of each word in the string (words that do not start with a letter remain unchanged). The optional first parameter, **xxx**, specifies the separators that you wish to use. The list must be enclosed in double quotes. If you want to use a double quote as a separator, prefix it with the [Escape Character](#).

**Examples:**

```
echo %@caps[" ",i love take command]
I Love Take Command

echo %@caps[",",peter,paul,mary]
Peter,Paul,Mary

echo %@caps[" ^","",sacrebleu!", he said]
"Sacrebleu!", He Said
```

#### 4.4.5.31 @CDROM

**@CDROM[d:]** : Returns **1** if the drive is an optical drive (CD-ROM, CD-RW, DVD, etc) or **0** otherwise. The drive letter must be followed by a colon.

**Examples:**

```
echo %@cdrom[C:]
0
```

```
echo %@cdrom[G:]
1
```

#### 4.4.5.32 @CEILING

**@CEILING[n]** : Returns the value of the smallest integer that is not less than *n*. @CEILING will perform an implicit [@EVAL](#) on its argument, so you can enter an arithmetic expression.

**Examples:**

```
echo %@ceiling[3.14]
4

echo %@ceiling[-3.14]
-3

echo %@ceiling[0]
0

echo %@ceiling[123*37.36]
4596
```

See also: [@FLOOR](#).

#### 4.4.5.33 @CHAR

**@CHAR[n]** : Returns the character corresponding to a Unicode numeric value. If the parameter is a set of numeric values, CHAR returns a string. For example %@CHAR[65] returns A; %@CHAR[65 66 67] returns ABC.

To display the non-ASCII Unicode characters (>= 128), you need to be using a Unicode font in **Take Command** and/or **TCC**.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

**Note:** Not all characters are printable. High ASCII characters (128-255) and Unicode characters may vary depending on the font used.

**Examples:**

```
echo %@char[65]
A

echo %@char[65 97 66 98 67 99]
AaBbCc
```

#### 4.4.5.34 @CLIP

**@CLIP[n]** : Returns line *n* from the Windows text clipboard. The first line is numbered 0. The string \*\*EOC\*\* is returned for all line numbers beyond the end of the clipboard.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal

format ("0x" followed by a sequence of 0-F hex digits).

**Examples:**

```
echo %@clip[0]

if "%@clip[2]" eq "***EOC***" echo No more data in the clipboard
```

#### 4.4.5.35 @CLIPW

**@CLIPW[string]** : Writes the *string* to the Windows text clipboard. Returns **0** if the operation was successful.

**Examples:**

```
if "%@clipw[save this line]" eq "0" echo Saved to the clipboard
Saved to the clipboard
```

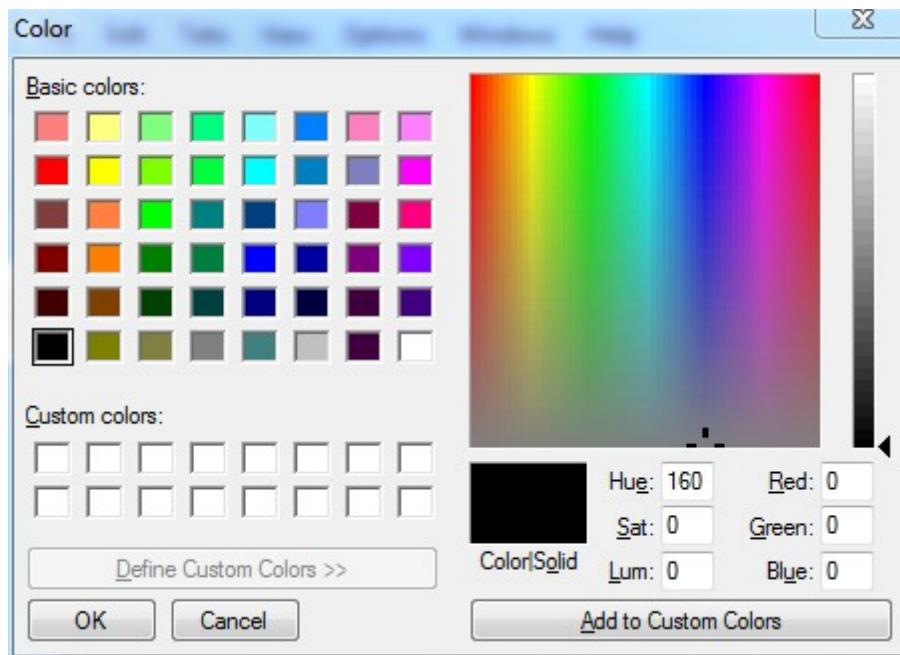
#### 4.4.5.36 @COLOR

**@COLOR[r,g,b]** : Displays the Windows color common dialog and returns the RGB value for the selected color as a string in the form **r,g,b** (e.g. **0,128,64**). To specify the initially selected color, use the **r** (red), **g** (green) and **b** (blue) parameters. If no parameters are provided, the initial selection will be black (**0,0,0**). The parameters are optional, but if one is used all three must be used.

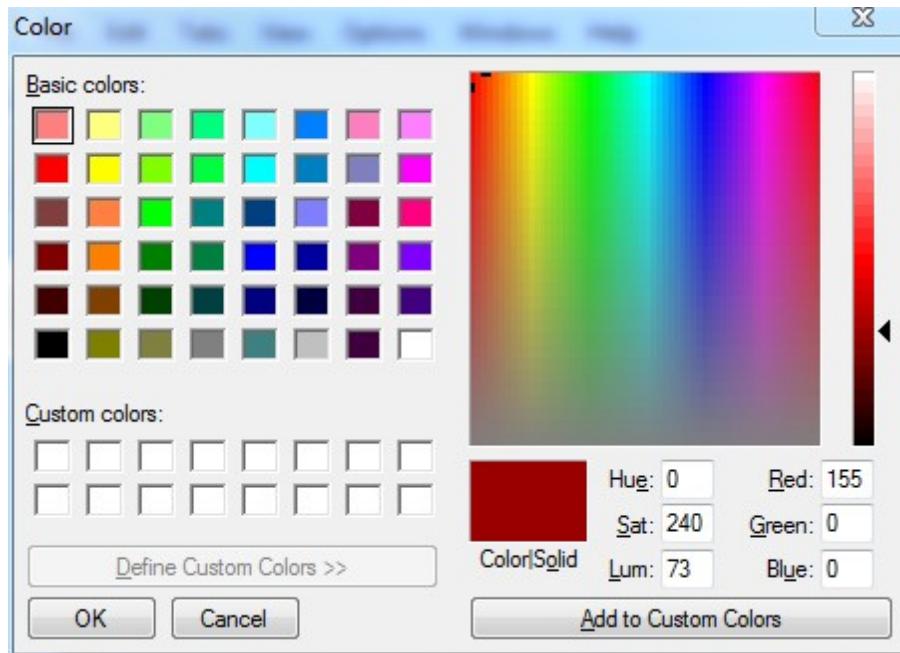
Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

**Examples:**

```
echo %@color[]
```



```
echo %@color[155,0,0]
```



#### 4.4.5.37 @COMMA

**@COMMA[n]** : Returns the number with commas (or the appropriate [Thousands character](#) for your current country setting) inserted where appropriate.

**Note:** Some [variable functions](#) can directly generate a numeric result with appropriate thousand separators if you add a **c** to their scale parameter.

**Examples:**

```
echo %@comma[12345678]
12,345,678
```

```
echo %@comma[0.12345678]
0.12345678
```

```
echo %@comma[%_pixels]
1,920
```

See also: [@CONVERT](#), [@FORMAT](#), [@FORMATN](#).

#### 4.4.5.38 @COMPARE

**@COMPARE[file1,file2]** : Returns **1** if the two files are identical, or **0** if they differ. @COMPARE supports FTP filenames for either *file1* or *file2*, but cannot compare two FTP files.

**Example:**

```
echo %@compare["c:\windows\system32\cmd.exe","c:\windows\syswow64\cmd.exe"]
0
```

#### 4.4.5.39 @CONSOLE

**@CONSOLE[title]** : Returns **1** if the specified window title belongs to a console window; **0** if it does not. The *title* may include [wildcards](#).

**Example:**

```
echo %@console[TCC Prompt]
1
```

#### 4.4.5.40 @CONSOLEB

**@CONSOLEB[handle]** - create or restore a console screen buffer. "Handle" is the handle to the desired screen buffer. If "handle" is -1, @CONSOLEB just returns the current buffer handle. If "handle" is 0, @CONSOLEB will create and activate a new console screen buffer. If "handle" is non-zero, @CONSOLEB will switch to that screen buffer. @CONSOLEB returns the handle to the active screen buffer. You can close an console handle with the @FILECLOSE function.

@CONSOLEB allows you to preserve the contents of the current screen buffer by switching to a second buffer temporarily and then back to the original buffer.

#### 4.4.5.41 @CONVERT

**@CONVERT[input, output, value]** : Returns a numeric string **value** converted from one number base (*input*) to another (*output*). Valid bases range from 2 to 36. The **value** can be between 0 and 2\*\*64-1. No error is returned if **value** is outside that range.

**Examples:**

```

echo binary 1010101 is decimal %@convert[2,10,1010101]
binary 1010101 is decimal 85

echo decimal 20 is hex %@convert[10,16,20]
decimal 20 is hex 14

echo hexadecimal FF is octal %@convert[16,8,FF]
hexadecimal FF is octal 377

echo this year is %@convert[10,2,%_year] in binary
this year is 11111011100 in binary

```

See also: [@COMMA](#), [@FORMAT](#), [@FORMATN](#).

#### 4.4.5.42 @COUNT

**@COUNT[c, string]** : Returns the number of times the character **c** appears in **string**.

**Examples:**

```

echo %@count[e,Another function example]
3

```

#### 4.4.5.43 @CRC32

String mode:   **@CRC32[s, string]**  
 File mode:   **@CRC32[[d,][f,]filename]**

Returns the CRC32 value (using the same algorithm as PKZIP or WINZIP) of the character in **string** or of the contents of the file **filename**. The first parameter must be **s** for a string, and any leading or trailing whitespace characters in **string** are included.

If the first argument for file mode is a **d**, @CRC32 will return the result in decimal (base 10) format. (This is the same format as POSIX 1003.2.) Otherwise, the result is returned in hexadecimal format.

**Filename** may be specified with or without an optional **f**. @CRC32 returns **-1** if the file does not exist, or it cannot be read.

Since **Take Command** handles all internal strings as Unicode, @CRC32 will return different results for a string and the identical string in an ASCII file.

See also: [@SHA256](#), [@SHA384](#), [@SHA512](#), and [@MD5..](#)

**Examples:**

```

echo %@crc32["C:\windows\explorer.exe"]
3F1E7CFE

echo %@crc32["%comspec"]
F36EB74C

echo %@crc32[d,"%comspec"]
4084119372

```

#### 4.4.5.44 @CWD

**@CWD[d:]** : Returns the current working directory of the specified disk drive in the format *d:\pathname*. If the current working directory is the root directory, the format is *d:\*. The drive letter must be followed by a colon.

**Examples:**

```
echo %@ cwd [C:]
c:\Windows

echo %@ cwd [%_disk:]
D:\release\version14
```

See also: [@CWDS](#).

#### 4.4.5.45 @CWDS

**@CWDS[d:]** : Returns the current working directory of the specified disk drive in the format *d:\pathname*. The drive letter must be followed by a colon.

**Examples:**

```
echo %@ cwds [C:]
c:\Windows\

echo %@ cwds [%_disk:]
D:\release\version14
```

See also: [@CWD](#).

#### 4.4.5.46 @DATE

**@DATE[date[,format]]** : Returns the number of days since January 1, 1980 for the specified date. See [date formats](#) for information on acceptable date formats. **Date** must be between 1980-01-01 and 2099-12-31 (inclusive).

**@DATE** accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Examples:**

```
echo %@ date [01-01-2012]
```

```
11688

echo %@date[2012-01-01,4]
11688

echo %@date[%_date]
11814
```

#### 4.4.5.47 @DATECONV

**@DATECONV[date,format]** - convert a date from the default format to another format. The output formats are:

- 0 system default
- 1 USA (mm/dd/yy)
- 2 European (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO 8601 (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Example:**

```
echo %@dateconv[5-1-2012,4]
2012-05-01
```

#### 4.4.5.48 @DAY

**@DAY[date[,format]]** : Returns the numeric day of the month for the specified date. See [date formats](#) for information on acceptable date formats.

**@DAY** accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Examples:**

```
echo %@day[2012-01-01]
1

echo %@day[%_date]
6
```

**4.4.5.49 @DEBUG**

**@DEBUG[*string*]** : Write a string to the system debugger.

**4.4.5.50 @DEC**

**@DEC[*string*]** : Returns :

- -1 if ***string*** is empty
- otherwise the same value as [@EVAL\[\*string\* - 1\]](#)

If ***string*** is the name of an environment variable, its value is used whether or not it is preceded by a percent sign % without modifying the value of the variable. To actually decrement the value of the variable **var** use:

```
set var=%@dec[%var]
```

**Example:**

```
set start=5
set result=%@dec[start]
echo %result
4
```

**4.4.5.51 @DECIMAL**

**@DECIMAL[*number*]**: Returns the portion of ***number*** to the right of the [Decimal character](#) as an integer numeric string. Trailing zeros are used to pad to the [Minimum Precision](#) specified for [@EVAL](#). For example:

```
%@decimal[%@eval[1/2]]
```

is 5 if minimum width is 0, and 50000 if minimum width is 5.

**@DECIMAL** will perform an implicit [@EVAL](#) on its argument, so you can enter an arithmetic expression (including the **@EVAL =min,max** format string following the argument).

**Examples:**

| <b>function</b>  | <b>value</b> |
|------------------|--------------|
| %@decimal[1234]  | 0            |
| %@decimal[1.234] | 234          |
| %@decimal[12.34] | 34           |

**4.4.5.52 @DESCRIPT**

**@DESCRIPT[*filename*]**: Returns the file description for the specified filename (see [DESCRIBE](#)). If there is no description for the file, **@DESCRIPT** returns an empty string.

The ***filename*** must be in quotes if it contains white space or special characters.

**Examples:**

```
echo %@descript["D:\My Path\Myfile.exe"]
```

```
echo %@descript[%comspec]
```

#### 4.4.5.53 @DEVICE

**@DEVICE[*name*]** : Returns **1** if the specified name is a character device (such as a serial port), or **0** if not. A trailing **:** is optional except for the pseudo-device CLIP: (to differentiate it from a possible filename named "clip").

**Examples:**

```
echo %@device[%comspec]
0

echo %@device[lpt1]
1

echo %@device[com1]
1

echo %@device[com5]
0

echo %@device[clip]
0

echo %@device[clip:]
1
```

#### 4.4.5.54 @DIGITS

**@DIGITS[*n*]**: Returns **1** if the string is composed of decimal digits only, otherwise it returns **0**. The [Decimal character](#), the [Thousands character](#), and the sign characters (+ or -) are not digits, and if they are present in the string @DIGITS will return **0**.

**Examples:**

```
echo %@digits[12345]
1

echo %@digits[-12345]
0

echo %@digits[1.2345]
0
```

#### 4.4.5.55 @DIRSTACK

**@DIRSTACK[*n*]** : Returns the name of the *n*th entry in the directory stack. The oldest is number 0. If no *n* parameter is specified, returns the total number of entries in the stack. The directory stack is set by calls to [PUSHD](#) / [POPD](#).

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal

format ("0x" followed by a sequence of 0-F hex digits).

See also: [DIRS](#), [POPD](#), [PUSHD](#) and [Directory Navigation](#).

**Examples:**

```
pushd c:\windows
pushd c:\windows\system32
echo %@dirstack[0]
C:\

echo %@dirstack[1]
C:\Windows

echo %@dirstack[]
2
```

#### 4.4.5.56 @DISKFREE

**@DISKFREE[d:[,scale[c]]]** : Returns the amount of free disk space on the specified drive. If you're specifying a drive, the drive letter must be followed by a colon. Optionally, you can specify a directory or UNC name, and @DISKFREE will display the free disk space on the drive referenced by that name (which may be different from the drive if the directory is a link to a directory on another drive).

The optional second parameter specifies the reporting scale (see [Memory Size / Disk Space / File Size Units and Report Format](#)). If the scale specification is suffixed with **c** the result will be formatted using the [thousands separator](#).

@DISKFREE supports [OpenAFS](#) names.

See also: [@DISKTOTAL](#) and [@DISKUSED](#).

**Examples:**

```
echo %@diskfree[c:]
19941240832

echo %@diskfree[%_disk:,Kc]
503,709,632
```

#### 4.4.5.57 @DISKTOTAL

**@DISKTOTAL[d:[,scale[c]]]** : Returns the total disk space on the specified drive. If you're specifying a drive, the drive letter must be followed by a colon. Optionally, you can specify a directory or UNC name, and @DISKTOTAL will display the total disk space on the drive referenced by that name (which may be different from the drive if the directory is a link to a directory on another drive).

The optional second parameter specifies the reporting scale (see [Memory Size / Disk Space / File Size Units and Report Format](#)). If the scale specification is suffixed with **c** the result will be formatted using the [thousands separator](#).

@DISKTOTAL supports [OpenAFS](#) names.

See also: [@DISKFREE](#) and [@DISKUSED](#).

**Examples:**

```
echo %@disktotal[c:]
120031539200
```

```
echo %@disktotal[%_disk:,Kc]
976,657,404
```

#### 4.4.5.58 @DISKUSED

**@DISKUSED[d:[,scale[c]]]** : Returns the amount of disk space in use on the specified drive. If you're specifying a drive, the drive letter must be followed by a colon. Optionally, you can specify a directory or UNC name, and @DISKUSED will display the disk space in use on the drive referenced by that name (which may be different from the drive if the directory is a link to a directory on another drive).

The optional second parameter specifies the reporting scale (see [Memory Size / Disk Space / File Size Units and Report Format](#)). If the scale specification is suffixed with **c** the result will be formatted using the [thousands separator](#).

@DISKUSED supports [OpenAFS](#) names.

See also: [@DISKFREE](#) and [@DISKTOTAL](#).

**Examples:**

```
echo %@diskused[c:]
100090298368
```

```
echo %@diskused[%_disk:,Kc]
472,947,772
```

#### 4.4.5.59 @DOMAIN

**@DOMAIN[name]** : Returns the domain of the computer specified by the DNS or NetBios **name**. If **name** is not specified, returns the domain of the local computer.

#### 4.4.5.60 @DOW

**@DOW[date[,format]]** : Returns the first three characters of the English name of the day of the week for the specified date ("Mon", "Tue", "Wed", etc.). See [date formats](#) for information on acceptable date formats.

@DOW accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)

- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Examples:**

```
echo %@dow[01-01-1980]
Tue
```

```
echo %@dow[%_date]
Sun
```

See also: [@IDOW](#).

**4.4.5.61 @DOWF**

**@DOWF[date[,format]]** : Returns the full English name of the day of the week for the specified date ("Monday", "Tuesday", etc.). See [date formats](#) for information on acceptable parameter formats.

@DOWF accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Examples:**

```
echo %@dowf[01-01-1980]
Tuesday
```

```
echo %@dowf[%_date]
Sunday
```

See also: [@IDOWF](#).

**4.4.5.62 @DOWI**

**@DOWI[date[,format]]** : Returns an integer representing the day of the week for the specified date (1 = Sunday, 2 = Monday, etc.). See [date formats](#) for information on acceptable date formats.

@DOWI accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Examples:**

```
echo %@dowi[01-01-1980]
3

echo %@dowi[%_date]
1
```

**4.4.5.63 @DOY**

**@DOY[date[,format]]** : Returns the day of year (1 - 366) for the specified date. See [date formats](#) for information on acceptable date formats.

@DOY accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Examples:**

```
echo %@doy[02-02-2010]
33

echo %@doy[%_date]
127
```

**4.4.5.64 @DRIVETYPE**

**@DRIVETYPE[drive]** : Return the type for the specified drive:

- 0 The drive type cannot be determined
- 1 The root path is invalid (no volume is mounted at the path)
- 2 Removable disk
- 3 Fixed disk
- 4 Remote (network) drive
- 5 CD-ROM
- 6 RAM disk

**Examples:**

```
echo %@drivetype[c:]
3

echo %@drivetype[z:]
4
```

```
echo %@drivetype[e:]
5
```

#### 4.4.5.65 @DRIVETYPEEX

**@DRIVETYPEEX[drive]** : Return the type for the specified drive:

- 0 The drive type cannot be determined
- 1 The root path is invalid (no volume is mounted at the path)
- 2 Removable disk
- 3 Fixed disk
- 4 Remote (network) drive
- 5 CD-ROM
- 6 RAM disk
- 7 DVD
- 8 Tape

**Examples:**

```
echo %@drivetypeex[c:]
3
```

```
echo %@drivetypeex[z:]
4
```

```
echo %@drivetypeex[e:]
7
```

#### 4.4.5.66 @ENUMSOURCES

**@ENUMSOURCES[n,server[,type]]** : Enumerate the servers on the network. **n** is the entry number in the list of servers (the first one is 0). **server** is the machine name(s) to match and it may contain [wildcards](#). Returns a null string if there are fewer than **n-1** matching servers. This function can be repeatedly called, incrementing **n** each time to enumerate all available server names until it returns a null string.

If **n** is -1, @ENUMSOURCES returns the number of matching servers.

@ENUMSOURCES takes an optional third argument to return only servers of that type. The possible types are:

- WORKSTATION - All workstations.
- SQLSERVER - Any server running Microsoft SQL Server
- DOMAIN - Primary domain controller
- DOMAINBACKUP - Backup domain controller
- DOMAIN\_ENUM - Primary domain
- LOCAL - Servers maintained by the browser
- AFP - Apple File Protocol servers
- TIME - Servers running the Timesource service
- PRINTQ - Server sharing print queue
- TERMINAL - Terminal Servers

CLUSTER - Server clusters in the domain  
 VSCLUSTER - Cluster virtual servers in the domain  
 MASTER - Server running the master browser service

**WARNING!** Windows may require a significant amount of time before returning data to this function when used on large networks.

**Examples:**

```
echo %@enumservers[0,L*]
\\LINKSTATION

for %i in (0 1 2) echo %@enumservers[%i,*]
\\LINKSTATION
\\MUSIC
\\WEBHOST
```

#### 4.4.5.67 @ENUMSHARES

**@ENUMSHARES[n,\\server\shares]** : Enumerate the share names for the specified server. **n** is the entry number in the list of shares (the first one is 0). **server** is the server name, and **shares** is the sharename(s) to match. **Shares** may contain [wildcards](#). Returns a null string if there are fewer than **n-1** matching shares. This function can be repeatedly called, incrementing **n** each time to enumerate all available shares until it returns a null string.

If the **n** is -1, @ENUMSHARES returns the number of matching sharenames.

**Examples:**

```
echo %@enumshares[0,\\LINKSTATION*]
\\LINKSTATION\info

for %i in (0 1 2) echo %@enumshares[%i,\\LINKSTATION*]
\\LINKSTATION\info
\\LINKSTATION\share
\\LINKSTATION\archive
```

#### 4.4.5.68 @ERRTEXT

**@ERRTEXT[n]** : Returns the operating system error text for the specified code. The text will be in the default language.

**Examples:**

```
echo %@errtext[2]
The system cannot find the file specified.

echo %@errtext[255]
The extended attributes are inconsistent.

echo %@errtext[%_syserr]
```

Incorrect function.

#### 4.4.5.69 @EVAL

**@EVAL[expression[=displayformat]]:** Evaluates a mathematical expression and returns its value in the format specified by **displayformat** or in the default format. [Parameter Interpretation](#) below describes what **expression** may contain. [Display precision and output format](#) below explains the result format.

The expression can contain environment variables and other variable functions, and may use any of the operators listed below. @EVAL also supports parentheses (to control evaluation order), commas, hexadecimals and decimal separators. Parentheses can be nested. @EVAL will strip leading and trailing zeros from the result unless you use the output formatting operators.

@EVAL supports very large numbers. The maximum size is 20,000 digits (10,000 digits to the left of the decimal point and 10,000 decimal places). If you want to use more than the default decimal values you'll need to change your [@Eval Precision](#) configuration options or use the "=x.y" format in [@EVAL](#). The integer-only operators (AND, OR, and XOR) are limited to 64-bit integers.

- ▶ [Parameter Interpretation](#)
- ▶ [Arithmetic operators](#)
- ▶ [Trigonometric and transcendental functions](#)
- ▶ [Other functions](#)
- ▶ [Order of precedence](#)
- ▶ [Precision of internal calculations](#)
- ▶ [Display precision and output format](#)
- ▶ [Examples](#)

#### Parameter Interpretation

**Expression** may contain environment and internal variables, [array variables](#) (not in **TCC/LE**), and variable functions. After all variables and functions have been expanded, it must be composed only of numeric strings and names of functions in [Trigonometric and transcendental functions](#) or [Other functions](#), connected by [Arithmetic operators](#) and optionally grouped with parentheses.

@EVAL permits you to simplify **expression** by dropping the % percent mark in front of the names of environment variables. This also prevents the **TCC** parser from expanding (possibly erroneously) variables before passing them to @EVAL. You must include % for internal variables and variable functions. @EVAL also permits you to use characters which normally have special meaning for **TCC** e.g., & < > ^ | without disabling their special meaning or quoting them.

**Note:** To ensure that **expression** is interpreted correctly, spaces should be placed on both sides of each operator, and parentheses used liberally. For example:

```
%@eval[(20 %% 3) + 4]
%@eval[12 and 65]
```

@EVAL also accepts numbers in the e exponent syntax; i.e. 1575e-2 = 15.75.

#### Number base

If a string starts with the characters **0x** it is interpreted as an integer in hexadecimal notation. If a string

starts with the characters **0b** it is interpreted as an integer in binary notation. Any other numeric string is considered to be a decimal number.

For example:

```
[c:\] echo %@eval[0x10 + 16]
32
```

You can specify hexadecimal output with the special syntax `@eval[...=H]`. For example:

```
echo %@eval[3*6=H]
```

will output 12 (hex). No leading 0x is included in the output. To convert between decimal and hexadecimal formats, see the [@CONVERT](#) function.

You can specify binary output with the special syntax `@eval[...=B]`. For example:

```
echo %@eval[3*6=B]
```

Hex and binary output is limited to 64-bit (signed) integers.

## Arithmetic operators

Every operator accepts both integer and non-integer parameters, except as noted below.

### Operators accepting fractional parameters

- +** (with one parameter) sign of numeric parameter (e.g. +3)
- +** (with two parameters) addition
- (with one parameter) negation of symbolic parameter (e.g., -n) or sign of numeric parameter (e.g. -1, +3)
- (with two parameters) subtraction
- \*** multiplication
- /** division
- \*\*** exponentiation
- !** boolean not

### Operators requiring integer parameters

- \** integer division (returns the integer part of the quotient)
- MOD** modulo (returns the remainder when the first parameter is divided by the second)
- %%** same as **MOD**
- SHL** arithmetic left shift of the first parameter, truncated toward zero to an integer, by the number of bits specified by the second parameter
- <<** same as **SHL**
- SHR** arithmetic right shift of the first parameter, truncated toward zero to an integer, by the number of bits specified by the second parameter
- >>** same as **SHR**
- >** greater than
- <** less than

### Operators which truncate parameters to integer

|              |                                                                                                                         |
|--------------|-------------------------------------------------------------------------------------------------------------------------|
| <b>AND</b>   | bitwise and (returns 1 for each bit position where the corresponding bits in both parameters are 1)                     |
| <b>&amp;</b> | same as <b>AND</b>                                                                                                      |
| <b>OR</b>    | bitwise or (returns 1 for each bit position where the corresponding bit in at least one parameter is 1)                 |
| <b> </b>     | same as <b>OR</b>                                                                                                       |
| <b>XOR</b>   | bitwise exclusive or (returns 1 for each bit position where the corresponding bits of the two parameters are different) |
| <b>^</b>     | same as <b>XOR</b>                                                                                                      |
| <b>~</b>     | unary NOT                                                                                                               |

### Trigonometric and transcendental functions

**Expression** may include the trigonometric and transcendental functions below. The argument is interpreted as radians.

|                 |                    |
|-----------------|--------------------|
| <b>log(x)</b>   | natural logarithm  |
| <b>log10(x)</b> | log 10             |
| <b>exp(x)</b>   | exponential        |
| <b>sin(x)</b>   | sine               |
| <b>asin(x)</b>  | arcsine            |
| <b>sinh(x)</b>  | hyperbolic sine    |
| <b>cos(x)</b>   | cosine             |
| <b>acos(x)</b>  | arccosine          |
| <b>cosh(x)</b>  | hyperbolic cosine  |
| <b>tan(x)</b>   | tangent            |
| <b>atan(x)</b>  | arctangent         |
| <b>tanh(x)</b>  | hyperbolic tangent |

The special string **PI** is a shortcut for the value **3.14159265358979323846**.

### Other functions (Not available in **TCC/LE**)

|                   |                                                                                |
|-------------------|--------------------------------------------------------------------------------|
| <b>abs(x)</b>     | absolute value                                                                 |
| <b>ceil(x)</b>    | ceiling                                                                        |
| <b>fact(x)</b>    | factorial                                                                      |
| <b>floor(x)</b>   | floor                                                                          |
| <b>gcd(x y)</b>   | greatest common divisor                                                        |
| <b>lcm(x y)</b>   | least common multiple                                                          |
| <b>ror(x y z)</b> | rotate <b>x</b> right <b>y</b> bits with a variable size of <b>z</b> (in bits) |
| <b>rol(x y z)</b> | rotate <b>x</b> left <b>y</b> bits with a variable size of <b>z</b> (in bits)  |

### Order of precedence

1. variables
2. expressions in matching parentheses
3. functions listed in [Trigonometric and transcendental functions](#)
4. exponentiation
5. multiplication, division, and MOD
6. addition and subtraction
7. **>**, **<**, AND, OR, XOR, NOT, SHL, and SHR

When multiple consecutive expressions of a single precedence level are used, evaluation is left to right.

For example, `3 + 4 * 2` will be interpreted as `3 + 8`, not as `7 * 2`. To change this order of evaluation, use parentheses to specify the order you want.

### Precision of internal calculations

`@EVAL` supports numbers up to 20,000 digits; it is highly unlikely you'll need greater precision than this!

### Display precision and output format

The maximum display precision is 10,000 digits to the left of the decimal point and 10,000 digits to the right. You can alter the default decimal precision with the [OPTION](#) command, the [@EVAL Precision](#) configuration options, and with the [SETDOS](#) /F command. You can change the decimal separator with the [decimal character](#) configuration option or the [SETDOS](#) /G command.

You can alter the display format for the current instance of `@EVAL` by specifying *displayformat*.

#### Hexadecimal display format

If *displayformat* is the letter **H**, output will be hexadecimal. If *displayformat* is **X**, the output will be hexadecimal with a leading **0x**.

#### Binary display format

If *displayformat* is the letter **B**, output will be binary.

#### Explicit precision

If *displayformat* is *i.a*, then:

- *i* must be a number which specifies the minimum decimal precision (the minimum number of decimal places displayed);
- *a* must be a number which sets the maximum decimal precision.
- the character separating *i* and *a* may be the comma if it is your decimal separator

You may specify either or both parameters *i* and *a*. If *i > a*, or if only *i* is specified, *i* is used as both the minimum and maximum precision, e.g. both `=2` and `=2.1` are equivalent to `=2.2`.

### Examples

| Expression                    | Value        |
|-------------------------------|--------------|
| <code>@eval[3 / 6=2.4]</code> | 0.50         |
| <code>@eval[3 / 6=4.4]</code> | 0.5000       |
| <code>@eval[3 / 7]</code>     | 0.4285714286 |
| <code>@eval[3 / 7=.4]</code>  | 0.4286       |
| <code>@eval[3 / 7=2.2]</code> | 0.42         |
| <code>@eval[3 / 7=2]</code>   | 0.42         |

See also: [@DEC](#) and [@INC](#).

#### 4.4.5.70 @EXEC

**@EXEC[command]** : Execute **command** and return its numeric exit code.

**Command** can be an alias, internal command, external command, **.BTM**, **.BAT**, or **.CMD** file.

By default, @EXEC returns the result code from **command** (see the [?](#) and [\\_?](#) variables). However, if in **command** you preface the command name with @ then @EXEC returns an empty string.

**Example:**

```
PROMPT=%@exec[@color 15 on %@if[%@removable[%_disk] eq 0,2,4] & echos [%_cwd%] & color 11 on 0]$s
```

See also: [@EXECSTR](#).

#### 4.4.5.71 @EXECARRAY

**@EXECARRAY[array,command]** : Execute the specified command and store the resulting lines in the specified [array variable](#). The array must be one-dimensional.

You must define the array before running @EXECARRAY. For example:

```
setarray aresult[10]
echo %@execarray[aresult,dir /u] >& nul
```

@EXECARRAY will read the number of lines specified in the array size definition, or the number of lines in the command output (whichever is less). @EXECARRAY returns the return value of the command.

The number of lines stored in the array is saved in the [EXECARRAY](#) internal variable.

#### 4.4.5.72 @EXECSTR

- **@EXECSTR[*n,command*]** : Runs the specified **command** and returns line *n* (or the first line if *n* is not specified) written to stdout by **command**. For example, to return the third line returned by VER /R:

```
echo %@execstr[2,command]
```

If *n* is negative, @EXECSTR starts at the last line and counts backwards.

(The /n option is not available in **TCC/LE**.)

@EXECSTR is useful for retrieving a result from an external utility. For example, if you have an external utility called **NETTIME.EXE** which retrieves the time of day from your network server and writes it to standard output, you could save it in an environment variable using a command like this:

```
set server_time=%@execstr[d:\path\nettme.exe]
```

If the same utility returned a result properly formatted for the TIME command, you could also use it to set the time on your system:

```
time %@execstr[d:\path\nettme.exe]
```

@EXECSTR can also be used with internal commands:

```
echo Newest file is: %@execstr[*dir /a:-d /h /o:-d /f]
```

**@EXECSTR** involves several extensive internal processing stages. You might be able to use more complex command sequences (pipes, command groups, etc.) as its parameter, but always *test* carefully first as the results may not always be what you expect. We recommend that you only use a single command (internal, external, batch file, etc.) parameter.

See also: [@EXEC](#) and [@EXECARRAY](#).

#### 4.4.5.73 @EXETYPE

**@EXETYPE[*filename*]**: Returns the application type for an executable file:

| <b>Code</b> | <b>Application type</b> |
|-------------|-------------------------|
| <b>0</b>    | Unknown                 |
| <b>1</b>    | DOS app                 |
| <b>2</b>    | PIF file                |
| <b>3</b>    | Win16                   |
| <b>4</b>    | Win 3.x VxD             |
| <b>5</b>    | OS/2                    |
| <b>6</b>    | Win32 GUI               |
| <b>7</b>    | Win32 console           |
| <b>8</b>    | Posix                   |

**Examples:**

```
echo %@exetype["dc:\windows\explorer.exe"]
6

echo %@exetype["%comspec"]
7
```

#### 4.4.5.74 @EXPAND

**@EXPAND[[*range...*] *filename*,[{+|-}]*rhsadecijopt*]** : Returns (in a single line), the names of all files and directories that are within the specified ***range*[*s*]**, AND match ***filename***, AND have the specified attributes. ***Filename*** may contain [wildcards](#) and [include lists](#). Returns an empty string if no files match. If the file list is longer than the allowed [command line length](#), it will be truncated without an error message. Each returned filename which contains white space or other special characters will be delimited by double quotes.

***Filename*** must be in double quotes if it contains white space or special characters.

The ***range*** and attribute parameters, if included, define properties of the files that will be included in the result as specified in [File Selection](#). Multiple ***range*** parameters may be included, but not more than one each of [description range](#), [size range](#), [date range](#), and [time range](#). ***Range*** parameters must precede ***filename***.

**Examples:**

```
echo %@expand[/[s2k,3k] *.txt]
```

displays all files with extension **txt** in the current directory with size at least 2000 bytes and at most 3000 bytes

```
echo %@expand[* ,d]
```

displays all subdirectories

```
echo %@expand[/ [d-365] %windir\w*.exe;w*.dll]
```

displays all files at most 365 days old in the Windows directory, with extension *EXE* or *DLL*, and name beginning with **W**.

#### 4.4.5.75 @EXT

**@EXT[filename]** : Returns the extension from *filename*, without a leading period. On volumes which support long file names, the extension can be up to 255 characters long. On FAT drives it can be up to 3 characters long. *filename* must be quoted if it contains white space or special characters.

On an LFN drive, the returned extension may contain white space or special characters. To avoid problems which could be caused by these characters, quote the returned extension before you pass it to other commands.

##### Examples:

```
set COMSPEC="c:\program files\jpsoft\tcmd14x64\tcc.exe"
echo %@ext[%@comspec]
exe

echo %@ext["LFN Names may have.very long extensions"]
very long extensions
```

#### 4.4.5.76 @FIELD

**@FIELD[["sep\_list",]n,string]** : Returns the *n*th field in *string*. The first field is numbered **0**. If *n* is negative, fields are counted backwards from the end of *string*. You can specify the rightmost field by setting *n* to -0.

You can specify a range of fields to return with the syntax:

```
@FIELD[["sep_list",]start[-end | +range],string]
```

Specify an inclusive range with a -. For example:

%@FIELD[2-4,A B C D E F G] will return "C D E". (Note that you cannot use inclusive ranges when starting from the end.)

You can specify a relative range with a +. For example:

%@FIELD[2+1,A B C D E F G] will return "C D".

The default list of separators for [@FIELD](#), [@FIELDS](#), [@WORD](#) and [@WORDS](#) consists of space, tab, and comma. You can use the optional first parameter, *sep\_list*, to specify the separators that you wish to use. If you want to use a double quote as a separator, prefix it with an [escape character](#), e.g., ^". Alphabetic characters in *sep\_list* are case sensitive. If you do not specify a separator list, @FIELD will skip any leading separators.

[@FIELD](#) and [@FIELDS](#) differ from [@WORD](#) and [@WORDS](#) in how multiple consecutive separators are counted. [@WORD](#) and [@WORDS](#) consider a sequence as a single separator, and ignore separators at either end of string. In contrast, [@FIELD](#) and [@FIELDS](#) count each occurrence of a separator individually, including those at either end of string.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits). To use hexadecimal form for a negative *n*, remember to use 32-bit 2's complement arithmetic, e.g., **0xFFFFFFFF** for -1. There is no hexadecimal form to specify field -0 (the rightmost field).

If *string* is double quoted, you must specify *sep\_list*.

See also: [@WORD](#), [@WORDS](#), [@FIELDS](#).

#### Examples:

| function                                         | value  |
|--------------------------------------------------|--------|
| %@field[2,zero,one,two,three]                    | two    |
| %@field[2,zero,,two,three]                       | two    |
| %@field["\",2,C:\Program Files\My Dir\myapp.exe] | My Dir |
| %@field[-2,zero,one,two,three]                   | one    |

#### 4.4.5.77 @FIELDS

**@FIELDS[["sep\_list"],]string** : Returns the number of fields in *string*.

The default list of separators for [@FIELD](#), [@FIELDS](#), [@WORD](#) and [@WORDS](#) consists of space, tab, and comma. You can use the optional first parameter, *sep\_list*, to specify the separators that you wish to use. If you want to use a quote mark as a separator, prefix it with an [escape character](#), e.g., ^". Alphabetic characters in *sep\_list* are case sensitive. If you do not specify a separator list, @FIELD will skip any leading separators.

[@FIELD](#) and [@FIELDS](#) differ from [@WORD](#) and [@WORDS](#) in how multiple consecutive separators are counted. [@WORD](#) and [@WORDS](#) consider a sequence as a single separator, and ignore separators at either end of string. In contrast, [@FIELD](#) and [@FIELDS](#) count each occurrence of a separator individually, including those at either end of string.

If *string* is double quoted, you must specify *sep\_list*.

#### Example:

```
echo %@fields[" ,",Now is the time]
```

4

See also: [@WORD](#), [@WORDS](#), [@FIELD](#).

#### 4.4.5.78 @FILEAGE

**@FILEAGE[filename[,a|c|w[u]]]** : Returns the date and time of the file as an age.

**Filename** must be in quotes if it contains white space or special characters. The optional second parameter selects which date field is returned for files on a VFAT or NTFS drive: **a** means the last access date, **c** means the creation date, and **w** means the last modification (write) date. The default is **w**.

If you append a **u** to the second argument, @FILEAGE will display the age in UTC.

**Examples:**

```
echo %@fileage["c:\windows\explorer.exe"]
129801709001110605
```

```
echo %@fileage["%comspec",c]
129801709001110605
```

See also: [Time Stamps](#), [@AGEDATE](#) and [@MAKEAGE](#).

#### 4.4.5.79 @FILEARRAY

**@FILEARRAY[array,filename]** : Read the specified file and store the resulting lines in the specified [array variable](#) (one line per element). The array must be one-dimensional.

You must define the array before running @FILEARRAY. For example:

```
setarray aresult[10]
echo %@filearray[aresult,test.dat]
```

@FILEARRAY will read the number of lines specified in the array size definition, or the number of lines in the files (whichever is less).

@FILEARRAY will return the number of lines read.

#### 4.4.5.80 @FILECLOSE

**@FILECLOSE[n]** : Closes the file whose handle is **n**. Returns **0** if the file was successfully closed, or **-1** if an error occurred.

This function should only be used with file handles returned by [@FILEOPEN!](#) If you use it with any other number you may damage other files opened by **TCC** (or by the program which started **TCC**).

See also the related handle-based functions:

|                             |                                                |
|-----------------------------|------------------------------------------------|
| <a href="#">@FILEOPEN</a>   | Open a file handle                             |
| <a href="#">@FILEREAD</a>   | Read next line from a file handle              |
| <a href="#">@FILESEEK</a>   | Move a file handle pointer                     |
| <a href="#">@FILESEEKL</a>  | Move a file handle pointer to a specified line |
| <a href="#">@FILEWRITE</a>  | Write next line to a file handle               |
| <a href="#">@FILEWRITEB</a> | Write data to a file handle                    |

[@TRUNCATE](#)

Truncate the file at the current position of the file handle pointer.

**Examples:**

```
set h=%@fileopen["d:\path\myfile.txt",write]
echo writing %@filewrite[%h,this is a test] bytes
echo closing handle #%h: %@fclose[%h]
```

**4.4.5.81 @FILEDATE**

**@FILEDATE[filename[,a|c|w[u,d]]]** : Returns the date a file was last modified, in the default country format (mm-dd-yy for the US), or as explicitly specified by the optional third parameter **d** (see [Date Display Formats](#)). **Filename** must be in quotes if it contains white space or special characters. The optional second parameter selects which date field is returned for files on an LFN drive: **a** means the last access date, **c** means the creation date, and **w** means the last modification (write) date, which is the default.

If you append a **u** to the second argument, @FILEDATE will display the date in UTC.

**Example:**

```
echo %@filedate["%comspec",c,4]
2012-04-29
```

See [Time Stamps](#), [@FILETIME](#), [@FILEAGE](#).

**4.4.5.82 @FILEHANDLE**

**@FILEHANDLE[handle]** : Returns the filename for the specified file handle (opened with [@FILEOPEN](#)).

**Example:**

```
set h=%@fileopen["d:\path\myfile.txt",r]
echo handle %h is : %@filehandle[%h]
handle 756 is : d:\path\myfile.txt
```

**4.4.5.83 @FILENAME**

**@FILENAME[filename]** : Returns the name and extension of a file, without a path.

The **filename** must be in quotes if it contains white space or special characters. On an LFN drive, the returned filename may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands.

**Examples:**

```
echo %@filename["D:\my path\myfile.exe"]
myfile.exe
```

```
echo %@filename["comspec"]
tcc.exe
```

#### 4.4.5.84 @FILEOPEN

**@FILEOPEN[*filename,r|read|w|rite|a|ppend|l,b|t|J*]** : Opens the file in the specified mode and returns the file handle as an integer. The optional third parameter controls whether the file is opened in binary or text mode. Text mode (the default) should be used to read text using [@FILEREAD](#) without a *length*, and to write text using [@FILEWRITE](#). Binary mode should be used to read binary data with [@FILEREAD](#) with a *length*, and to write binary data with [@FILEWRITEB](#). Returns -1 if the file cannot be opened.

**Filename** must be in quotes if it contains white space or special characters. To read from standard input, use **CON:** for the filename.

To open a file for both reading and writing, open it in append mode, then use [@FILESEEK](#) to position to the start of the file (or any other desired location) before performing additional operations.

@FILEOPEN can also open named pipes. The pipe name must begin with `\.\pipe\`. @FILEOPEN first tries to open an existing pipe; if that fails it tries to create a new pipe. Pipes are opened in blocking mode, duplex access, byte-read mode, and are inheritable. @FILEOPEN will not return until another process connects to the pipe. For more information on named pipes see your Windows documentation.

@FILEOPEN can open file streams on NTFS drives if the stream name is specified. See [NTFS File Streams](#) for additional details on file streams.

You must reference the file exclusively using the returned file handle, and you must close the file using the file handle. This is especially important when you are debugging a batch program which uses @FILEOPEN. If you suspect that file handles have been opened and not closed, you should restart **TCC**.

#### Examples:

```
set h=%@fileopen["d:\path\myfile.txt",write]
echo writing %@filewrite[%h,this is a test] bytes
echo closing handle #%h: %@fileclose[%h]
```

See also the related handle-based functions:

|                             |                                                                       |
|-----------------------------|-----------------------------------------------------------------------|
| <a href="#">@FILECLOSE</a>  | Close a file handle                                                   |
| <a href="#">@FILEREAD</a>   | Read next line from a file handle                                     |
| <a href="#">@FILESEEK</a>   | Move a file handle pointer                                            |
| <a href="#">@FILESEEKL</a>  | Move a file handle pointer to a specified line                        |
| <a href="#">@FILEWRITE</a>  | Write next line to a file handle                                      |
| <a href="#">@FILEWRITEB</a> | Write data to a file handle                                           |
| <a href="#">@TRUNCATE</a>   | Truncate the file at the current position of the file handle pointer. |

#### 4.4.5.85 @FILEREAD

**@FILEREAD[*n[,length]*]** : Reads data from the file whose handle is *n*. Returns the string **\*\*EOF\*\*** if you attempt to read past the end of the file. If *length* is not specified, @FILEREAD will read until the next CR or LF (end of line) character. If *length* is specified, @FILEREAD will read *length* bytes regardless of any

end of line characters.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

This function should only be used with file handles returned by [@FILEOPEN](#). If you use it with any other number you may damage other files opened by **TCC** (or by the program which started **TCC**).

Beware of characters with special meaning to **Take Command**, such as redirection and piping symbols, within the file. Use [SETDOS](#) /X with appropriate codes as needed.

#### Example:

```
set h=%@fileopen["d:\path\myfile.txt",r]
echo reading %@fileread[%h,32]
echo closing handle #%h: %@fileclose[%h]
```

See also the related handle-based functions:

|                             |                                                                       |
|-----------------------------|-----------------------------------------------------------------------|
| <a href="#">@FILECLOSE</a>  | Close a file handle                                                   |
| <a href="#">@FILEOPEN</a>   | Open a file handle                                                    |
| <a href="#">@FILESEEK</a>   | Move a file handle pointer                                            |
| <a href="#">@FILESEEKL</a>  | Move a file handle pointer to a specified line                        |
| <a href="#">@FILEWRITE</a>  | Write next line to a file handle                                      |
| <a href="#">@FILEWRITEB</a> | Write data to a file handle                                           |
| <a href="#">@TRUNCATE</a>   | Truncate the file at the current position of the file handle pointer. |

#### 4.4.5.86 @FILEREADB

**@FILEREADB[n,length]** : Reads **n** bytes of data from the file whose handle is **n**. Returns the string \*\*EOF\*\* if you attempt to read past the end of the file. The data will be returned as a string of space-separated numeric digits representing the ASCII value of each character.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

This function should only be used with file handles returned by [@FILEOPEN](#). If you use it with any other number you may damage other files opened by **TCC** (or by the program which started **TCC**).

Beware of characters with special meaning to **TCC**, such as redirection and piping symbols, within the file. Use [SETDOS](#) /X with appropriate codes as needed.

#### Example:

```
set h=%@fileopen["d:\path\myfile.txt",r]
echo reading %@filereadb[%h,32]
echo closing handle #%h: %@fileclose[%h]
```

See also the related handle-based functions:

|                            |                            |
|----------------------------|----------------------------|
| <a href="#">@FILECLOSE</a> | Close a file handle        |
| <a href="#">@FILEOPEN</a>  | Open a file handle         |
| <a href="#">@FILESEEK</a>  | Move a file handle pointer |

[@FILESEEKL](#)  
[@FILEWRITE](#)  
[@FILEWRITEB](#)  
[@TRUNCATE](#)

Move a file handle pointer to a specified line  
 Write next line to a file handle  
 Write data to a file handle  
 Truncate the file at the current position of the file handle pointer.

#### 4.4.5.87 @FILES

**@FILES**[/**S**[[+]**n**]] [**range...**][/**H**] **filename**[,[{+|-}]**rhsadecijoptJ**] : Returns the number of files within **range** that match **filename** and have the specified attributes. **Filename** may contain [wildcards](#) and [include lists](#). Returns 0 if no files match. To check files in multiple directories use @FILES once for each, and add the results with [@EVAL](#).

**Filename** must be in double quotes if it contains white space or special characters.

The **range** and [attribute](#) parameters, if included, define properties of the files that will be included in the result as specified in [File Selection](#). Multiple **range** parameters may be included, but not more than one each of [description range](#), [size range](#), [date range](#), and [time range](#). **Range** parameters must precede **filename**. [Exclusion ranges](#) are not supported.

If you include the optional **/S** argument, @FILES will search the current directory and all of its subdirectories for matching files. If you specify a number after the **/S**, @FILES will limit the subdirectory recursion to that number. (For example, if you have a directory tree "\a\b\c\d\c", **/S2** will only affect the "a", "b", and "c" directories.)

If you specify a + followed by a number after the **/S**, @FILES will not count any files until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\c, **/S+2** will not count anything in \a or \a\b. (Not available in **TCC/LE**.)

If you include the optional **/H** argument, @FILES will not include the **"."** and **".."** directory entries in the count.

#### Examples:

```
echo %@files[/[s2k,3k] *.txt]
 number of files with extension txt in the current directory with size at least 2000 bytes and at most 3000 bytes

echo %@files[* ,d]
 number of subdirectories

echo %@files[/[d-365] %windir\w*.exe;w*.dll]
 number of files at most 365 days old in the Windows directory, with extension EXE or DLL, and name beginning with w
```

#### 4.4.5.88 @FILESEEK

**@FILESEEK**[**n,offset,start**] Moves the file pointer of the file whose handle is **n** by **offset** bytes from the reference location specified via **start** (see the table below). The return value of @FILESEEK is the offset of the file pointer from the beginning of the file after the specified move. If **offset** is negative, the file pointer is moved from the reference location toward the beginning of the file. If **offset** is positive, the file pointer is moved from the reference location toward the end of the file. If **offset** is 0, the pointer is moved

to the reference location.

If the function fails, the return value is **-1**.

| <b>start</b> | <b>reference location</b> |
|--------------|---------------------------|
| <b>0</b>     | beginning of file         |
| <b>1</b>     | current file pointer      |
| <b>2</b>     | end of file               |

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

This function should only be used with file handles returned by [@FILEOPEN](#). If you use it with any other number you may damage other files opened by **TCC** (or by the program which started **TCC**).

### Useful special cases

If you set **offset** to **0** :

- `@FILESEEK[n,0,0]` moves the file pointer to the beginning of file
- `@FILESEEK[n,0,1]` returns the current location of the file pointer without moving it.
- `@FILESEEK[n,0,2]` moves the file pointer to the end of file, and returns the current file size.

### Example:

```
set h=%@fileopen["d:\path\myfile.txt",rw]
echo file size = %@fileseek[%h,0,2]
echo closing handle #%h: %@fileclose[%h]
```

See **also** the related handle-based functions:

|                             |                                                                       |
|-----------------------------|-----------------------------------------------------------------------|
| <a href="#">@FILECLOSE</a>  | Close a file handle                                                   |
| <a href="#">@FILEOPEN</a>   | Open a file handle                                                    |
| <a href="#">@FILEREAD</a>   | Read next line from a file handle                                     |
| <a href="#">@FILESEEKL</a>  | Move a file handle pointer to a specified line                        |
| <a href="#">@FILEWRITE</a>  | Write next line to a file handle                                      |
| <a href="#">@FILEWRITEB</a> | Write data to a file handle                                           |
| <a href="#">@TRUNCATE</a>   | Truncate the file at the current position of the file handle pointer. |

#### 4.4.5.89 [@FILESEEKL](#)

**@FILESEEKL[n,line[,1]]** : Moves the file pointer to the specified *line* in the open file whose handle is *n*. The first line in the file is numbered **0**. Returns the new position of the pointer, in bytes from the start of the file. The third parameter is optional, and determines the starting point for the seek. If not specified, or set to a value other than **1**, `@FILESEEKL` starts at the beginning of the file. If set to **1**, `@FILESEEKL` will start from the current position in the file.

If the function fails, the return value is **-1**.

`@FILESEEKL` must read each line of the file up to the target line in order to position the pointer, and can therefore cause significant delays if used in a loop or on a large file.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

This function should only be used with file handles returned by [@FILEOPEN](#). If you use it with any other number you may damage other files opened by **TCC** (or by the program which started **TCC**).

#### Example:

```
set h=%@fileopen["d:\path\myfile.txt",rw]
echo file line 10 = %@fileseekl[%h,10,2]
echo closing handle %h: %@fileclose[%h]
```

See also the related handle-based functions:

|                             |                                                                       |
|-----------------------------|-----------------------------------------------------------------------|
| <a href="#">@FILECLOSE</a>  | Close a file handle                                                   |
| <a href="#">@FILEOPEN</a>   | Open a file handle                                                    |
| <a href="#">@FILEREAD</a>   | Read next line from a file handle                                     |
| <a href="#">@FILESEEK</a>   | Move a file handle pointer                                            |
| <a href="#">@FILEWRITE</a>  | Write next line to a file handle                                      |
| <a href="#">@FILEWRITEB</a> | Write data to a file handle                                           |
| <a href="#">@TRUNCATE</a>   | Truncate the file at the current position of the file handle pointer. |

#### 4.4.5.90 @FILESIZE

**@FILESIZE[/S|[+|n]] [range...] filename[,scale[c][,a]]** : Returns the size of a file, or -1 if the file does not exist. If **filename** includes [wildcards](#) or an [include list](#), it returns the combined size of all matching files. The optional third parameter **a** tells **@FILESIZE** to return the amount of space allocated for the file(s) on the disk. (Network drives and compressed drives may not always report allocated sizes accurately, depending on the way the network or disk compression software is implemented.)

**Filename** must be in quotes if it contains white space or special characters.

The second parameter specifies the reporting scale (see [Memory Size / Disk Space / File Size Units and Report Format](#)). Adding the letter **c** requests the result be formatted using the [thousands separator](#).

The optional **range** parameter defines properties of the files that will be included in the result as specified in [File Selection](#). Multiple **range** parameters may be included, but not more than one each of [description range](#), [size range](#), [date range](#), and [time range](#). **Range** parameters must precede **filename**. [Exclusion ranges](#) are not supported.

If you include the optional **/S** argument, **@FILESIZE** will search the current directory and all of its subdirectories for matching files. If you specify a number after the **/S**, **@FILESIZE** will limit the subdirectory recursion to that number. For example, if you have a directory tree "\a\b\c\d\le", **/S2** will only affect the "a", "b", and "c" directories. (Not available in **TCC/LE**.)

If you specify a **+** followed by a number after the **/S**, **@FILESIZE** will not count any file sizes until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\le, **/S+2** will not count anything in \a or \ab.

#### Examples:

```
echo %@filesize[d:\path\myfile.ext]
```

```
echo %@filesize["%comspec",bc]
359,400

echo %@filesize["%comspec",bc,a]
360,448
```

#### 4.4.5.91 @FILETIME

**@FILETIME[filename,[a|c|w|u]][,s]]** : Returns the time of day a file was last modified, in hh:mm format. **Filename** must be in quotes if it contains white space or special characters. The optional second parameter selects which time field is returned for files on an LFN drive: **a** means the last access time, **c** means the creation time, and **w** means the last modification (write) time, which is the default. Times are normally returned with hours and minutes only. To retrieve seconds as well, add **s** as the optional third parameter. On non-NTFS drives, the last access time is always returned as 00:00, and without a seconds field (see [Time Stamp](#) for additional details).

If you append a **u** to the second argument, @FILETIME will display the time in UTC.

#### Examples:

```
echo %@filetime["D:\my path\myfile.exe"]
16:40

echo %@filetime["%comspec",c,s]
11:01:40
```

See also: [@FILEDATE](#), [@FILEAGE](#).

#### 4.4.5.92 @FILEWRITE

**@FILEWRITE[n,text]**: Writes a line to the file whose handle is **n**. Returns the number of characters written, or -1 if an error occurred. A CR/LF will be appended to **text**.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

This function should only be used with file handles returned by [@FILEOPEN](#). If you use it with any other number you may damage other files opened by **TCC** (or by the program which started **TCC**).

Beware of characters with special meaning to **TCC**, such as redirection and piping symbols, within the file. Use [SETDOS /X](#) with appropriate codes as needed.

#### Example:

```
set h=%@fileopen["d:\path\myfile.txt",w]
echo writing %@filewrite[%h,32]
echo closing handle #%h: %@fileclose[%h]
```

See also the related handle-based functions:

|                                    |                                                                       |
|------------------------------------|-----------------------------------------------------------------------|
| <a href="#"><u>@FILECLOSE</u></a>  | Close a file handle                                                   |
| <a href="#"><u>@FILEOPEN</u></a>   | Open a file handle                                                    |
| <a href="#"><u>@FILEREAD</u></a>   | Read next line from a file handle                                     |
| <a href="#"><u>@FILESEEK</u></a>   | Move a file handle pointer                                            |
| <a href="#"><u>@FILESEEKL</u></a>  | Move a file handle pointer to a specified line                        |
| <a href="#"><u>@FILEWRITEB</u></a> | Write data to a file handle                                           |
| <a href="#"><u>@TRUNCATE</u></a>   | Truncate the file at the current position of the file handle pointer. |

#### 4.4.5.93 @FILEWRITEB

**@FILEWRITEB[n,length,string]** : Writes the specified number of bytes from the *string* to the file whose handle is *n*. Returns the number of bytes written, or -1 if an error occurred.

**Note:** Writes ASCII output when passed a Unicode string. Note that if you're trying to write non-English (>128) characters with @FILEWRITEB, the output will probably not match the input.

If the *length* argument is -1, @FILEWRITEB will read the string argument as a series of ASCII values in decimal or hex to write to the file. For example:

```
echo %@filewriteb[%file,-1,0xe0 0xF2 0xA9]
```

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

This function should only be used with file handles returned by [@FILEOPEN!](#). If you use it with any other number you may damage other files opened by **TCC** (or by the program which started **TCC**).

Beware of characters with special meaning to **TCC**, such as redirection and piping symbols, within the file. Use [SETDOS](#) /X with appropriate codes as needed.

#### Example:

```
set h=%@fileopen["d:\path\myfile.txt",r]
echo writing %@filewriteb[%h,10,Write some characters from this string]
echo closing handle %h: %@fclose[%h]
```

See also the related handle-based functions:

|                                   |                                                                      |
|-----------------------------------|----------------------------------------------------------------------|
| <a href="#"><u>@FILECLOSE</u></a> | Close a file handle                                                  |
| <a href="#"><u>@FILEOPEN</u></a>  | Open a file handle                                                   |
| <a href="#"><u>@FILEREAD</u></a>  | Read next line from a file handle                                    |
| <a href="#"><u>@FILESEEK</u></a>  | Move a file handle pointer                                           |
| <a href="#"><u>@FILESEEKL</u></a> | Move a file handle pointer to a specified line                       |
| <a href="#"><u>@FILEWRITE</u></a> | Write next line to a file handle                                     |
| <a href="#"><u>@TRUNCATE</u></a>  | Truncate the file at the current position of the file handle pointer |

#### 4.4.5.94 @FILTER

**@FILTER[chars,string]** : Removes any characters in "string" that aren't in "chars".

#### Example:

To remove all non-numeric characters from a variable:

```
set var=abc1234
echo %@filter[0123456789,%var]
1234
```

#### 4.4.5.95 @FINDCLOSE

**@FINDCLOSE[filename]:** Signals the end of a [@FINDFIRST](#) ... [@FINDNEXT](#) sequence. You must use this function to release the directory search handle. **Filename** is unnecessary, this function can be simply called as %@FINDCLOSE[] without parameters. @FINDCLOSE returns 0 if a [@FINDFIRST](#) ... [@FINDNEXT](#) sequence is in effect, a non-zero value otherwise.

**Examples:**

```
echo %@findfirst[*.*]
echo %@findclose[]
```

#### 4.4.5.96 @FINDFIRST

**@FINDFIRST[/[range...]] filename,[,+|-]rhsadecijopt]** : Returns the name of the first file that matches **filename**, which may include [wildcards](#) and/or an [include list](#), and which file has the properties specified in the optional [range](#) and [attribute](#) parameters.

**Filename** must be in quotes if it contains white space or special characters.

The **range** and attribute parameters, if included, define properties of the files that will be included in the search as specified in [File Selection](#). Multiple **range** parameters may be included, but not more than one each of [size](#), [date](#), [time](#), and [file exclusion](#). **Range** parameters must precede **filename**. Each **range** parameter is of the form

```
/ [a . . .]
```

where **a** is one of **d**, **s**, **t**, and/or **!**, followed by the range parameters.

On an LFN drive, the returned filename may contain white space or other special characters. Unlike [@EXPAND\[\]](#), no double quotes are added by this function. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. See the notes under [Variable Functions](#) for additional details.

**@FINDFIRST[]** locates the *first* file matching the requirements. To find more matching files, you must use [@FINDNEXT\[\]](#), and terminate the search with [@FINDCLOSE\[\]](#).

**Warning:** @FINDFIRST searches may not be nested!

**Examples:**

```
%@findfirst/[d-30] *
locate files created no more than 30 days ago
```

```
%@findfirst/[s2k,3k] "%windir*.exe",a
locate files with the extension exe, the archive flag set, and at least 2,000 bytes but not more than 3,000 bytes long, in the Windows directory.
```

#### 4.4.5.97 @FINDNEXT

**@FINDNEXT[*filename*, [ ][-rhsadecijopt]]**: Returns the name of the next file that matches the filename(s) in the previous @FINDFIRST call. Returns an empty string when no more files match. @FINDNEXT should only be used after a successful call to [@FINDFIRST](#).

You do not need to include the ***filename*** parameter, because it must be the same as the one used in the previous @FINDFIRST call, unless you want to change the file attributes for @FINDNEXT. ***Filename***, if used, must be in quotes if it contains white space or special characters.

The attribute parameter, if included, defines the attributes of the files that will be included in the search as specified in [Attribute Switches](#).

**Range** parameters may not be used in this function. The ***range*** parameters specified in the preceding @FINDFIRST call remain effective.

If you don't need to change the attribute parameters established by the preceding @FINDFIRST, you can simply use this function as %@FINDNEXT without parameters.

On an LFN drive, the returned filename may contain white space or other special characters. Unlike [@EXPAND](#), no double quotes are added by this function. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. See the notes under [Variable Functions](#) for additional details.

@FINDFIRST[] locates the first file matching the requirements. To find more matching files, you must use [@FINDNEXT](#), and terminate the search with [@FINDCLOSE](#).

#### Examples:

```
echo %@findfirst[*]
echo %@findnext[]
echo %@findnext[* ,d]
echo %@findclose[]
```

#### 4.4.5.98 @FLOOR

**@FLOOR[*n*]**: Returns the largest integer that is not greater than ***n***. @FLOOR will perform an implicit [@EVAL](#) on its argument, so you can enter an arithmetic expression.

#### Examples:

```
echo %@floor[3.14]
3

echo %@floor[-3.14]
-4

echo %@floor[0]
0

echo %@floor[123]
```

---

123

See also: [@CEILING](#).

#### 4.4.5.99 @FOLDERS

**@FOLDERS[/S[[+]n]] [range...] dirname[,{+|-}rhsadecijopt]** : Returns the number of folders (subdirectories) within **range** that match **dirname** and have the specified attributes. **Dirname** may contain [wildcards](#) and [include lists](#). Returns 0 if no folders match. To check folders in multiple source directories use @FOLDERS once for each, and add the results with [@EVAL](#).

**Dirname** must be in double quotes if it contains white space or special characters.

The **range** and [attribute](#) parameters, if included, define properties of the folders that will be included in the result as specified in [File Selection](#). Multiple **range** parameters may be included, but not more than one each of [description range](#), [date range](#), and [time range](#). **Range** parameters must precede **dirname**. [Exclusion ranges](#) are not supported.

If you include the optional **/S** argument, @FOLDERS will search the current directory and all of its subdirectories for matching folders. If you specify a number after the **/S**, @FOLDERS will limit the subdirectory recursion to that number. (For example, if you have a directory tree "\a\b\c\d\c", **/S2** will only affect the "a", "b", and "c" directories.)

If you specify a **+** followed by a number after the **/S**, @FOLDERS will not count any files until it gets to that depth in the subdirectory tree. For example, if you have a directory tree \a\b\c\d\c, **/S+2** will not count anything in \a or \a\b.

If you are searching for subdirectories (i.e., by specifying "d" in the attribute argument), @FOLDERS will not count the **"."** and **".."** directory entries.

**Example:**

```
echo %@folders[c:\windows]
58

echo %@folders[/s,c:\windows]
17728
```

#### 4.4.5.100 @FONT

**@FONT[n]** : Returns console font information. **n** is the type of information requested:

- 0 - font name (Windows usually returns an empty string unless you've previously set the font)
- 1 - font width
- 2 - font height
- 3 - font weight
- 4 - font family
- 5 - font index in console font table

**Examples:**

```
echo %@font[0]
Consolas

echo %@font[1]
6

echo %@font[2]
12

echo %@font[3]
700

echo %@font[4]
54

echo %@font[5]
6
```

**4.4.5.101 @FORMAT**

**@FORMAT[format,string]** : Reformats *string*, truncating it or padding it with spaces or zeros as necessary. **format** is of the format **[*-j*.*a*]**. If the optional minus sign is present, the result is left justified; otherwise it is right justified. If *i* is specified, and its first digit is **0**, the padding character will be **0**, otherwise it will be a space. *i* is the minimum number of characters in the result, *a* is the maximum number of characters. If *a* is less than *i*, it will be ignored.

If *string* doesn't exist, @FORMAT treats it as an empty string and pads the output accordingly.

**Examples**

| function             | value   |
|----------------------|---------|
| "%@format[7>Hello]"  | "Hello" |
| "%@format[.3>Hello]" | "Hel"   |
| "%@format[4,5]"      | " 5"    |
| "%@format[04,5]"     | "0005"  |
| "%@format[-04,5]"    | "5000"  |

See also: [@COMMA](#), [@CONVERT](#), [@FORMATN](#).

**4.4.5.102 @FORMATN**

**@FORMATN[[-]*width*[.*precision*],*value*]** : Formats a numeric value. **Width** is a nonnegative integer specifying the minimum number of characters printed. If **Width** has a leading **0**, the number will be left-padded with zeros. If the number of characters in the output value is less than the specified width, blanks are added to the left or the right of the values depending on whether the "**-**" flag (for left alignment) is specified, until the minimum width is reached. **Precision** specifies the number of digits after the decimal point. The **value** is rounded to the appropriate number of digits.

If you don't specify a precision, @FORMATN will default to 16 decimal places, and may not round the

number appropriately. (For example, @FORMATN[3,3.4] will produce "3.399999999999999".)

#### Examples:

```
echo %@formatn[5.10,%@eval[2300*4.7]]
10810.0000000000
```

```
echo %@formatn[010.3,5]
000005.000
```

See also: [@COMMA](#), [@CONVERT](#), [@FORMAT](#), [@FORMATNC](#).

#### 4.4.5.103 @FORMATNC

**@FORMATNC[[-]width[,precision],value]** : Formats a numeric value and automatically inserts the thousands separator. **Width** is a nonnegative integer specifying the minimum number of characters printed. If **Width** has a leading 0, the number will be left-padded with zeros. If the number of characters in the output value is less than the specified width, blanks are added to the left or the right of the values depending on whether the "-" flag (for left alignment) is specified, until the minimum width is reached. **Precision** specifies the number of digits after the decimal point. The **value** is rounded to the appropriate number of digits.

#### Examples:

```
echo %@formatnc[5.10,%@eval[2300*4.7]]
10,810.0000000000
```

```
echo %@formatnc[010.3,5]
000005.000
```

See also: [@COMMA](#), [@CONVERT](#), [@FORMAT](#), [@FORMATN](#).

#### 4.4.5.104 @FSTYPE

**@FSTYPE[d:]** : Returns the file system type for the specified drive or sharename. @FSTYPE returns NTFS for a drive that uses the Windows NTFS file system. It returns FAT32 for FAT32 drives, and FAT for FAT12, FAT16, and VFAT drives.

You can specify either a drive name or a UNC name.

#### Examples:

```
echo %@fstype[c:]
NTFS
```

```
echo %@fstype[e:]
FAT32
```

```
echo %@fstype[\\"Music\\iTunes"]
NTFS
```

#### 4.4.5.105 @FTYPE

**@FTYPE[*xxx*[,*u*]]** : Returns the open command string for the specified file type. @FTYPE looks first in ...\\SHELL\\OPEN2\\COMMAND, then (if no match was found) in ...\\SHELL\\OPEN\\COMMAND. If the optional second argument *u* is specified, @FTYPE will look in HKCU\\SOFTWARE\\CLASSES.

Example:

```
echo %@ftype[Word.Document.8]
"C:\Program Files\Microsoft Office\Office14\WINWORD.EXE" /n ""
```

See also [@ASSOC](#) and [FTYPE](#).

#### 4.4.5.106 @FULL

**@FULL[*filename*]** : Returns the full path and filename of a file. *Filename* must be in quotes if it contains white space or special characters. On an LFN drive, the returned filename may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. See the notes under [Variable Functions](#) for additional details.

**Note:** The @FULL function makes no assumption about the existence of a file or directory. The *filename* parameter can be any string and the function will attempt to turn it into a fully qualified "volume + path + name" specification, whether that full reference exists or not.

Examples:

```
cdd c:\\windows
echo %@full[explorer.exe]
C:\\Windows\\explorer.exe

echo "%@full[.]"
"C:\\Windows"

echo "%@full[\"\\Program Files\"]"
"C:\\Program Files"
```

#### 4.4.5.107 @FUNCTION

**@FUNCTION[*name*]** : Returns the definition of the specified [user-defined function](#) *name* as a string, or a null string if the function doesn't exist. When manipulating strings returned by @FUNCTION you may need to disable certain special characters with [SETDOS /X](#). Otherwise, command separators, redirection characters, and other similar punctuation in the function may be interpreted as part of the current command, rather than part of a simple text string.

Example:

```
echo %@function[myfunction]
```

See the [FUNCTION](#) command.

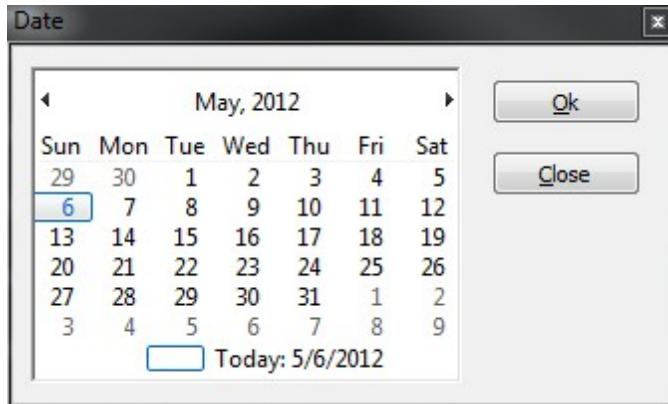
#### 4.4.5.108 @GETDATE

**@GETDATE[/date]** : Display a calendar dialog and returns the selected date in *yyyy-mm-dd* format.

You can optionally pass a default date (also in *yyyy-mm-dd* format). If you do not specify a default date, **@GETDATE** will use the current date.

**Example:**

```
echo %@getdate[]
```



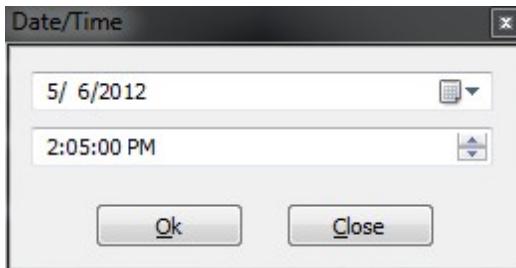
#### 4.4.5.109 @GETDATETIME

**@GETDATETIME[/date time]** : Display a date/time picker dialog and returns the selected date in *yyyy-mm-dd hh:mm:ss* format.

You can optionally pass a default date and time (also in *yyyy-mm-dd hh:mm:ss* format). If you do not specify a default date, **@GETDATETIME** will use the current date and time.

**Example:**

```
echo %@getdatetime[]
```



#### 4.4.5.110 @GETDIR

**@GETDIR[d:\path[,title]]** : Pops up a dialog box to select a directory. **d:\path** specifies the initial directory; if it is not specified, **@GETDIR** defaults to the current directory. Returns the chosen directory as a string, or an empty string if the user selects "Cancel" or presses Esc.

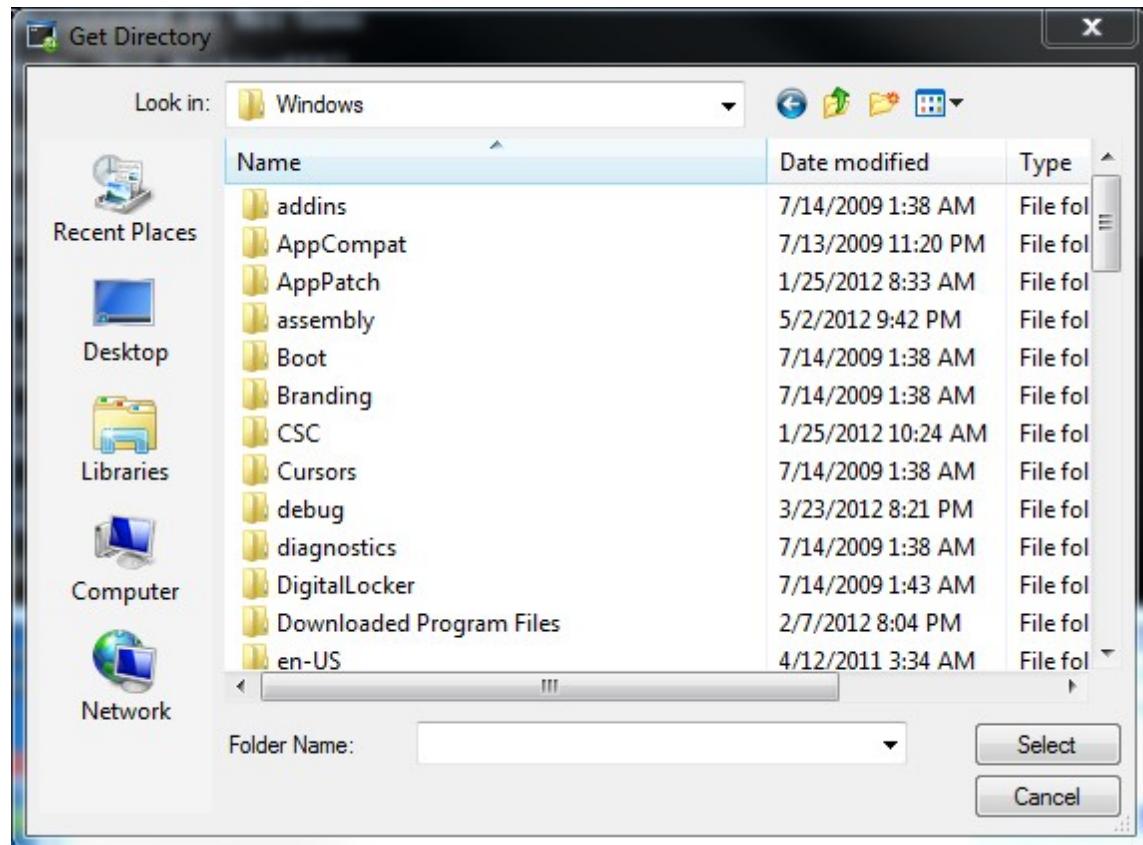
**d:\path** must be in quotes if it contains white space or special characters. On an LFN drive, the returned path may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned path before you pass it to other commands. See the notes under

[Variable Functions](#) for additional details.

@GETDIR accepts an optional second parameter to set the title of the dialog box.

**Example:**

```
cd %@getdir["C:\windows"]
```



**Note:** @GETDIR deals with directories. All directories are folders, but not all folders are directories. To select a symbolic folder, see [@GETFOLDER](#).

#### 4.4.5.111 @GETFILE

**@GETFILE[d:\path\filename[,filter[,title]]]**: Pops up a dialog box to select a file. **d:\path\filename** specifies the initial directory and filename shown in the dialog, and may include wildcards. Returns the full path and name of the selected file or an empty string if the user selects "Cancel" or presses Esc. The optional second parameter specifies the file extension to use. You can specify multiple extensions by separating them with semicolons. For example, **%@getfile[c:\windows,\*.exe;\*.btm]** lets the user select from .EXE and .BTM files only.

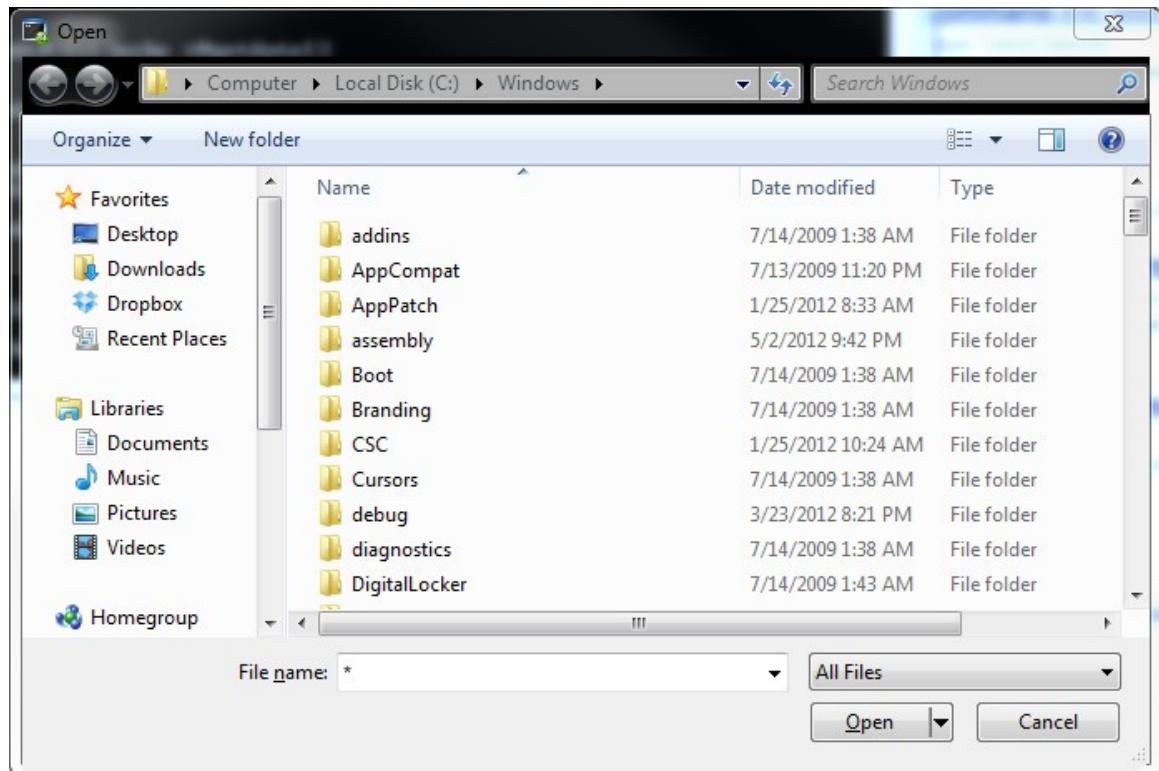
The parameters must be in quotes if they contain white space or special characters. On an LFN drive, the returned filename may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. See the notes under [Variable Functions](#) for additional details.

@GETFILE accepts an optional third parameter to set the title of the dialog box.

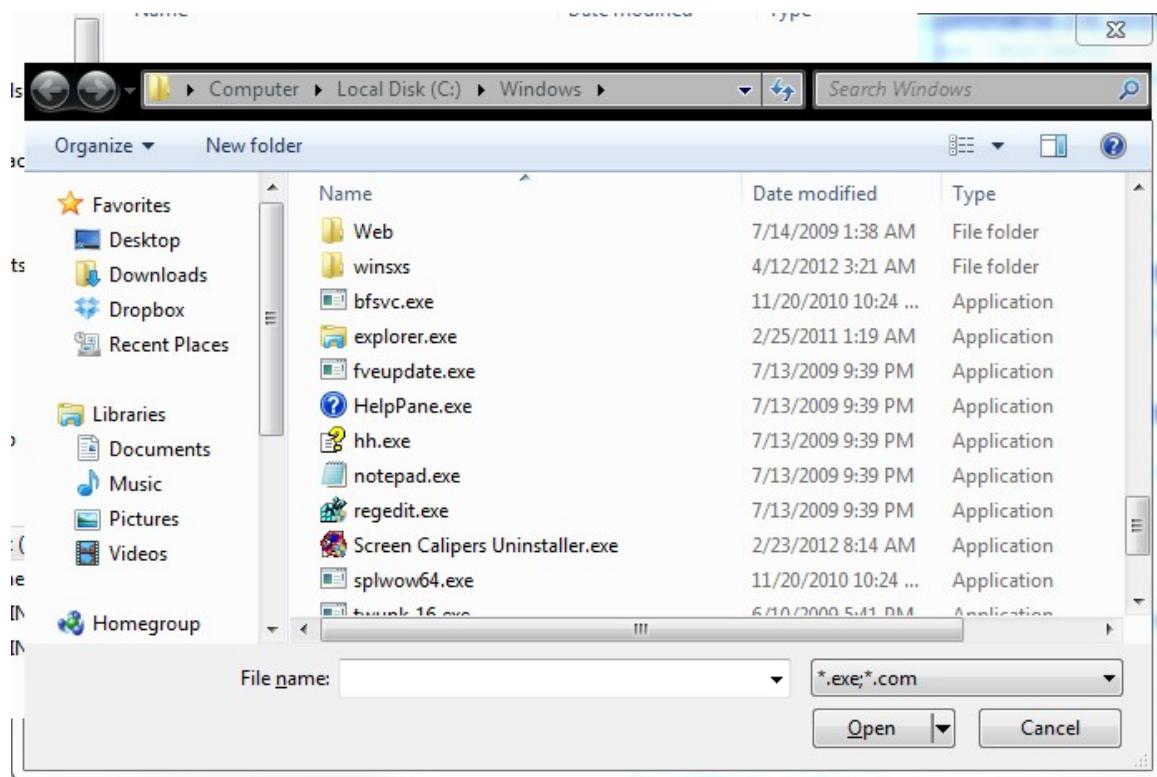
If you're looking for directories, use [@GETFOLDER](#).

**Examples:**

```
echo %@getfile[*]
```



```
echo %@getfile[%windir*,*.exe]
```



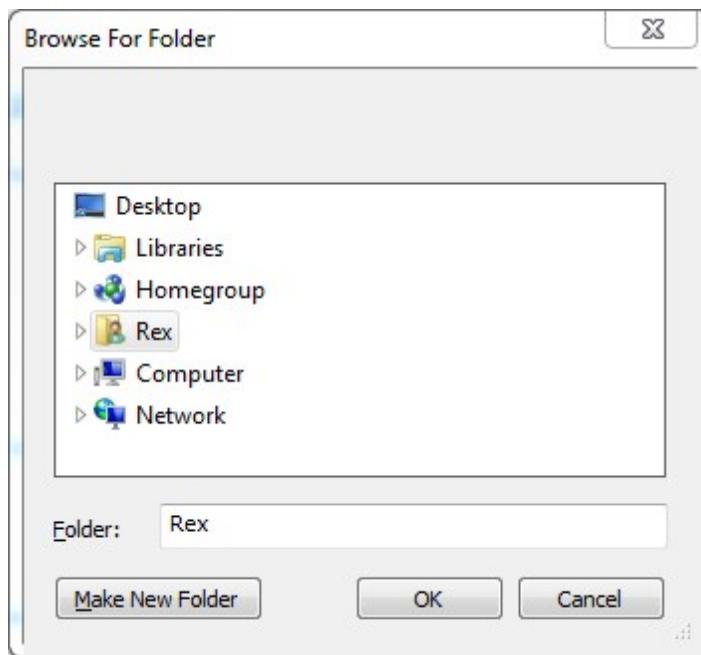
#### 4.4.5.112 @GETFOLDER

**@GETFOLDER[*startdir[,title]*]** : Returns a folder selected from a tree view of available symbolic folders. If you don't specify a start folder, @GETFOLDER starts at **My Computer** or the equivalent symbolic folder in your Windows configuration.

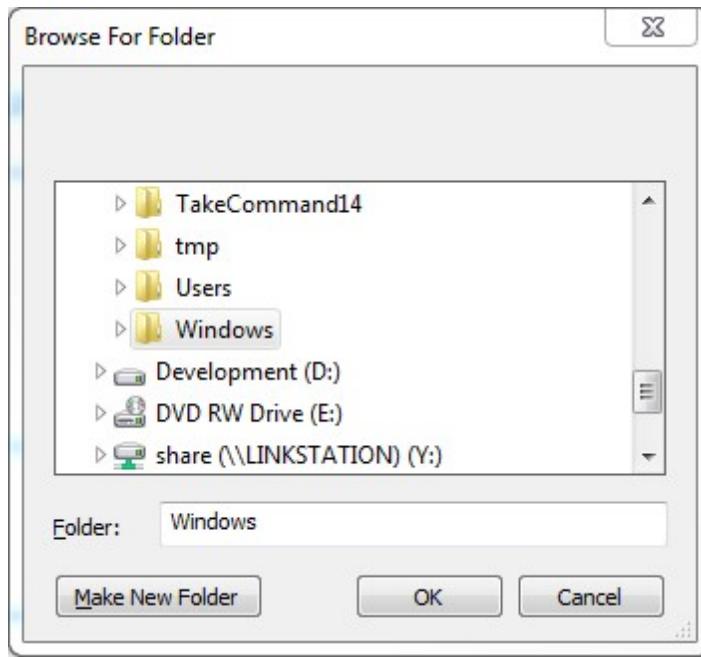
The optional second argument sets the text to display above the tree view.

##### Examples:

```
echo %@getfolder[]
```



```
echo %@getfolder["c:\windiws"]
```



**Note:** @GETFOLDER deals with folders. All directories are folders, but not all folders are directories. To select a directory, see [@GETDIR](#).

#### 4.4.5.113 @GROUP

**@GROUP[server,group,user]** : Returns 1 if **user** is a member of the specified **group**. **server** specifies the DNS or NetBIOS name of the computer on which the function is to execute.

#### 4.4.5.114 @HEXDECODE

**@HEXDECODE[s,string]** : Create a text string from a hexadecimal input string. Returns the text string.  
**@HEXDECODE[inputfile,outputfile]** : Decode a hex encoded file. Returns 0 if the output file was successfully written.

**Example:**

```
echo %@hexdecode[s,656e636f6465207468697320737472696e67]
decode this string
```

#### 4.4.5.115 @HEXENCODE

**@HEXENCODE[s,string]** : Create a hexadecimal string from a text input string. Returns the hex string.  
**@HEXENCODE[inputfile,outputfile]** : Encode a text file as a hex encoded file. Returns 0 if the output file was successfully written.

**Example:**

```
echo %@hexencode[s,encode this string]
656e636f6465207468697320737472696e67
```

#### 4.4.5.116 @HISTORY

**@HISTORY[x,y]** : Returns a line or word from the [command history](#). (This function will prove most useful in keystroke aliases). **x** is the line to retrieve (current line = 0), and **y** is the specific word (first word = 0) desired within that line. If **y** is omitted, @HISTORY returns the entire line.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

#### 4.4.5.117 @HTMLDECODE

**@HTMLDECODE[string]** : Decode an HTML string. The HTML escaped characters (i.e., &gt;) are replaced with their original values.

**Example:**

```
echo "%@htmldecode[This is & a string]"
"This is & a string"
```

See also [TPIPE](#).

#### 4.4.5.118 @HTMLENCODE

**@HTMLENCODE[string]** : Encode a string for HTML, replacing characters like > < & with the HTML escaped characters (i.e., &gt; for >).

**Example:**

```
echo "%@htmlecode[This is & a string]"
"This is & a string"
```

See also [TPIPE](#).

#### 4.4.5.119 @IDOW

**@IDOW[date[,format]]** : Returns the 3-character abbreviation for the day of the week for the specified date, in the current locale language. See [date formats](#) for information on date formats.

@IDOW accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)

**Examples:**

```
echo %@idow[01-01-1980]
Tue
```

```
echo %@idow[%_date]
Sun
```

See also: [@DOW](#).

#### 4.4.5.120 @IDOWF

**@IDOWF[date[,format]]** : Returns the full name for the day of the week for the specified date, in the current locale language. See [date formats](#) for information on date formats.

@IDOWF accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)

**Examples:**

```
echo %@idowf[01-01-2010]
Tuesday
```

```
echo %@idowf[%_date]
Sunday
```

See also: [@DOWF](#).

#### 4.4.5.121 @IF

**@IF[condition,string1,string2]**: Evaluates **condition** according to the rules described in [Conditional Expressions](#), and if **true**, it returns **string1**, otherwise it returns **string2**. Leading and trailing white space in **string1** and **string2** is retained. Either string may be empty or contain white space only. **WARNING:**

Both **string1** and **string2** are evaluated whether or not used. Do not use @IF if evaluating either one of the strings may fail; use the **IF** or **IFF** command instead.

### Examples

- 1) The expression

```
%@IF[2 == 2,Correct!,Oops!]
```

returns **Correct!**

- 2) The command

```
echo Good %@if[%_hour ge 12,evening,morning]!
```

displays **Good morning!** in the AM hours and **Good evening!** in the PM hours.

- 3) Assuming **A** and **C** are files in the current directory, but **B** is a subdirectory, the command:

```
for %x in (A B C) echo "%x" is %@if[isfile "%x", ,not] a file
```

will display

```
"A" is a file
"B" is not a file
"C" is a file
```

### 4.4.5.122 @INC

**@INC[string]** returns

- 1 if **string** is empty
- otherwise the same value as [@EVAL\[string + 1\]](#)

If **string** is the name of an environment variable, its value is used whether or not it is preceded by a percent mark % without modifying the value of the variable. To actually increment the value of the variable **var** use

```
set var=%@inc[%var]
```

**Example:**

```
set start=5
set result=%inc[start]
echo %result
6
```

### 4.4.5.123 @INDEX

**@INDEX[string1,string2[,n]]**: Returns the offset of **string2** within **string1**, or -1 if **string2** is not found or if **string1** is empty. The first or leftmost position in **string1** is numbered 0. The optional third parameter **n** has three different interpretations:

If **n > 0**, it specifies that the **n**th match from left to right is desired.

If **n** < 0 or it is prefixed with the minus sign -, it specifies that the -**n**th match from right to left is desired.

If **n**=0, the total number of matches is desired.

When **n** is omitted, the value returned is the offset of the *first* (leftmost) match.

## Tips

- searching for a **comma** :

1. quote **string1** (to prevent the expected comma making it appear as more than one parameter)
2. use [escape character](#) or its pseudovariable form %= in **string2** to escape the comma  

```
echo %@index["TCC, Take Command, TCCL",^,,2]
```

- searching for a **double quote** :

1. use [escape character](#) or its pseudovariable form %= in **string2** to escape the double quote
2. use the special form ^q to represent it in **string2**:

```
echo %@index[contains a "quoted" word,^q,0]
```

See [Codes for Escapable Characters](#) for details.

## Examples:

In all examples below

- **string1:** This is a fine help file
- **string2:** h

| <b>n</b>       | <b>result</b> | <b>purpose</b>          |
|----------------|---------------|-------------------------|
| <b>omitted</b> | 1             | locate leftmost         |
| 0              | 2             | count occurrences       |
| 1              | 1             | locate leftmost         |
| 2              | 15            | locate second leftmost  |
| 3              | -1            | locate third leftmost   |
| -1             | 15            | locate rightmost        |
| -2             | 1             | locate second rightmost |
| -3             | -1            | locate third rightmost  |

## 4.4.5.124 @INIREAD

**@INIREAD[file,section,entry]**: Returns the value of the first matching **entry** from the specified **file**, or an empty string if either **file** or the entry in **file** does not exist. If **file** contains more than one section named **section**, only the first one is searched for **entry**.

**File**, **section**, and **entry** must be in quotes if they contain white space or special characters.

### File selection

Both the name and extension of **file** must be specified. This function does not apply a default extension. If **file** does not explicitly include a path, @INIREAD uses %Windir%, the Windows installation directory. To

use the current directory, you must explicitly specify it, e.g., using .\ as the path.

### Example

```
%@iniread[c:\tcmd\tcmd.ini,TakeCommand,history]
```

returns the size of the command history if it is specified in *TCMD.INI*.

#### 4.4.5.125 @INIWRITE

**@INIWRITE[*file*,*section*,*entry*,*string*]**: Creates, updates, or deletes an entry in the specified *file*. If *file* does not exist, it will be created. @INIWRITE returns 0 for success or -1 for failure.

*File*, *section*, and *entry* must be in quotes if they contain white space or special characters.

##### File selection

Both the name and extension of *file* must be specified. This function does not apply a default extension. If *file* does not explicitly include a path, @INIWRITE uses %Windir%, the Windows installation directory. To use the current directory, you must explicitly specify it, e.g., using .\ as the path.

##### Action

If *file* does not exist, it will be created. If *string* is empty, *file* will be empty, otherwise a section line and a directive line will be created.

The remaining descriptions relate to the case when *file* exists.

If more than one match for *section* exists in *file*, only the first one is searched for *entry*. If more than one match exists for *section* and *entry*, only the first match is one is affected. Searching starts at the beginning of the file, and stops on the first match.

If *string* is empty, the matching *entry*, if any, is deleted. If *string* is not empty, and there is a matching *section* and *entry*, it is modified. If *string* is not empty, and there is no matching *section* and *entry*, it is created.

If *entry* is empty, the matching *section* (if any) is deleted.

##### Examples

```
echo %@iniwrite[c:\tcmd\tcmd.ini,TakeCommand,history,8192]
```

will set the size of the command history to 8,192 bytes.

```
echo %@iniwrite[c:\tcmd\tcmd.ini,TakeCommand,history,]
```

will remove the *history* entry from the file.

#### 4.4.5.126 @INODE

**@INODE[*filename*]** : Returns the inode (in hex) for the specified file.

When files are hard-linked to one another (see [MKLNK](#)), they share the same inode.

`@INODE` may not work for remote files (depending on your network redirector and the type of server you are querying).

#### Example:

```
echo %@inode[c:\windows\explorer.exe]
```

```
00040000:000199D3
```

#### 4.4.5.127 @INSERT

`@INSERT[offset,string1,string2]` : Inserts **string1** into **string2** starting at **offset**. The first offset in **string2** is **0**. If **offset** is greater than the length of **string2**, **string1** will be appended to the end of **string2**. If **offset** is negative, its value is used to count backward from the end of **string2** (but not past its beginning). Setting **offset** to **-0** is the same as setting it to **0**, i.e., **string1** will precede **string2** in the result. To include a comma in **string1**, precede it with your [escape character](#).

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits). To use hexadecimal form for a negative **offset**, remember to use 32-bit 2's complement arithmetic.

Examples:

| function                                           | value                 |
|----------------------------------------------------|-----------------------|
| <code>%@insert[1,arm,wing]</code>                  | warming               |
| <code>%@insert[8,very ,this is useful]</code>      | this is very useful   |
| <code>%@insert[255,^, very!,this is useful]</code> | this is useful, very! |
| <code>%@insert[-9,very ,this is useful]</code>     | this very is useful   |
| <code>%@insert[0,abcde,xyz]</code>                 | abcdexyz              |

#### 4.4.5.128 @INSTR

`@INSTR[start,[length],string]` : Returns a substring, beginning at offset **start** and continuing for **length** characters. If **length** is positive or it is omitted, the offset is measured from the beginning (i.e., left end) of the string. If **length** is omitted, all of the **string** beginning at offset **start** is returned. If **length** is negative, the offset is measured leftward from the right end of the string, and its length is specified by the value of **length** without the minus sign. [@SUBSTR](#) is an older version of the same function.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits). To use hexadecimal form for a negative **length**, remember to use 32-bit 2's complement arithmetic.

Examples:

| function                                             | value               |
|------------------------------------------------------|---------------------|
| <code>=%@instr[8,3,this is useful]=</code>           | =use=               |
| <code>=%@instr[8,,this is useful]=</code>            | =useful=            |
| <code>=%@instr[8,-4,this is useful]=</code>          | =is u=              |
| <code>=%@instr[8,,commas, they don't matter]=</code> | =they don't matter= |

#### 4.4.5.129 @INT

**@INT[n]:** Returns the integer part of the number *n*. @INT will perform an implicit [@EVAL](#) on its argument, so you can use an arithmetic expression for *n*.

**Examples:**

```
echo %@int[1234]
1234
```

```
echo %@int[1.234]
1
```

```
echo %@int[12.34]
12
```

#### 4.4.5.130 @IPADDRESS

**@IPADDRESS[hostname]:** Returns the numeric IP address for the specified hostname. The result is displayed in the standard format nnn.nnn.nnn.nnn. An invalid or unknown hostname will return an error (see [@ERRTEXT](#) to decipher the error number if necessary).

See also [@IPNAME](#).

Example:

```
echo %@ipaddress[jpsoft.com]
141.101.124.120
```

#### 4.4.5.131 @IPADDRESSN

**@IPADDRESSN[n]:** Returns the IP address of the adapter at index *n*

Example:

```
echo %@ipaddressn[0]
192.168.1.25
```

#### 4.4.5.132 @IPALIASES

**@IPALIASES[name]:** Returns the other names (if any) corresponding to the host with the specified name.

#### 4.4.5.133 @IPDESC

**@IPDESC[n]:** Returns the description for the adapter at index *n*

**Example:**

```
echo %@ipdesc[0]
NETGEAR WNA3100 N300 Wireless USB Adapter
```

#### 4.4.5.134 @IPDHCP

**@IPDHCP[n]:** Returns the DHCP server for the adapter at index *n*

**Example:**

```
echo %@ipdhcp[0]
192.168.1.254
```

#### 4.4.5.135 @IPGATEWAY

**@IPGATEWAY[n]:** Returns the gateway for the adapter at index *n*

**Example:**

```
echo %@ipgateway[0]
192.168.1.1
```

#### 4.4.5.136 @IPIPV6

**@IPIPV6N[n]:** Returns the IPv6 address of the adapter at index *n*

**Example:**

```
echo %@ipipv6n[0]
fe80::e432:b8ae:c538:4191
```

#### 4.4.5.137 @IPNAME

**@IPNAME[*numeric\_IP*]** : Returns the host name for the specified *numeric\_IP* address. An IP address **0** returns the name of the current local host (usually the computer name). The IP address can be expressed in the common format **nnn.nnn.nnn.nnn** or as a packed decimal. An invalid or unknown IP address returns an error (see [@ERRTEXT](#) to decipher the error number if necessary).

See also: [@IPADDRESS](#).

**Examples:**

```
echo %@ipname[173.194.43.40]
lga15s35-in-f8.1e100.net
```

```
echo %@ipname[0]
ASUS-PC
```

#### 4.4.5.138 @IPNAMEN

**@IPNAMEN[n]**: Returns the name of the adapter at index *n*

**Example:**

```
echo %@ipnamen[0]
{E80FB8C8-0218-3B34-1188-528293717098}
```

#### 4.4.5.139 @IPPHYSICAL

**@IPPHYSICAL[n]**: Returns the physical address of the adapter at index *n*

**Example:**

```
echo %@ipphysical[0]
30-4a-92-3e-66-1b
```

#### 4.4.5.140 @IPPORT

**@IPPORT[service]** : Returns the port number for the specified service.

#### 4.4.5.141 @IPSERVICEALIASES

**@IPSERVICEALIASES[service]** : Returns the aliases for the specified service.

#### 4.4.5.142 @IPSUBNET

**@IPSUBNET[n]** : Returns the subnet mask of the adapter at index *n*.

**Example:**

```
echo %@ipsubnet[0]
255.255.255.0
```

#### 4.4.5.143 @IPTYPE

**@IPTYPE[n]** : Returns the type of the adapter at index *n*. Possible values include:

OTHER  
WIRELESS  
ETHERNET  
TOKENRING

FDDI  
PPP  
LOOPBACK  
SLIP

**Example:**

```
echo %@iptype[0]
WIRELESS

echo %@iptype[1]
ETHERNET
```

**4.4.5.144 @IPWINS**

**@IPWINS[n]** : Returns 1 if the adapter at index *n* uses WINS.

**Example:**

```
echo %@ipwins[0]
0
```

**4.4.5.145 @IPWINSSERVER**

**@IPWINSSERVER[n]** : Returns the primary WINS Server for the adapter at index *n*.

**Example:**

```
echo %@ipwinsserver[0]
```

**4.4.5.146 @IPZONEID**

**@IPZONEID[n]** : Returns the IPv6 Zone ID (also known as a scope ID) for the adapter at index *n*. The values of the Zone ID are defined relative to the sending host

**Example:**

```
echo %@ipzoneid[0]
```

**4.4.5.147 @ISALNUM**

**@ISALNUM[string]**: Returns **1** if *string* is entirely composed of alphabetic (a-z, A-Z) and/or numeric (0 - 9) characters; **0** otherwise.

See also: [@ISALPHA](#), [@ISASCII](#), [@ISCNTRL](#), [@ISDIGIT](#), [@ISPRINT](#), [@ISPUNCT](#), [@ISSPACE](#), [@ISXDIGIT](#).

**Example:**

```
echo %@isalnum[123abc]
```

```
1

echo %@isalnum[123 abc]
0

echo %@isalnum[123.456]
0
```

#### 4.4.5.148 @ISALPHA

**@ISALPHA**[*string*]: Returns **1** if *string* is entirely composed of alphabetic (a-z, A-Z) characters; **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISCNTRL](#), [@ISDIGIT](#), [@ISPRINT](#), [@ISPUNCT](#), [@ISSPACE](#), [@ISXDIGIT](#).

**Example:**

```
echo %@isalpha[abc]
1

echo %@isalpha[ABC]
1

echo %@isalpha[A B C]
0
```

#### 4.4.5.149 @ISASCII

**@ISASCII**[*string*]: Returns **1** if *string* is entirely composed of 7-bit ASCII characters (0x00 - 0x7F); **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISCNTRL](#), [@ISDIGIT](#), [@ISPRINT](#), [@ISPUNCT](#), [@ISSPACE](#), [@ISXDIGIT](#).

**Examples:**

```
echo %@isascii[abc]
1

echo %@isascii[abc 123]
1

echo %@isascii["abc"⌚]
0
```

#### 4.4.5.150 @ISCNTRL

**@ISCNTRL[*string*]**: Returns **1** if *string* is entirely composed of ASCII control characters (0x00 - 0x1F or 0x7F); **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISASCII](#), [@ISDIGIT](#), [@ISPRINT](#), [@ISPUNCT](#), [@ISSPACE](#), [@ISXDIGIT](#).

**Examples:**

```
echo %@iscntrl [abc]
0

set var=^r^n
echo %@iscntrl [%var]
1
```

#### 4.4.5.151 @ISDIGIT

**@ISDIGIT[*string*]** : Returns **1** if *string* is entirely composed of decimal digits (0- 9); **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISASCII](#), [@ISCNTRL](#), [@ISPRINT](#), [@ISPUNCT](#), [@ISSPACE](#), [@ISXDIGIT](#).

**Example:**

```
echo %@isdigit[0]
1

echo %@isdigit[123.456]
0

echo %@isdigit[-123]
0
```

#### 4.4.5.152 @ISFLOAT

**@ISFLOAT[*string*]** : Returns **1** if *string* is composed only of numeric characters, a decimal separator, and an optional sign and/or thousands separator(s). The decimal separator and thousands separator are determined by your default country settings.

**Example:**

```
echo %@isfloat[1234]
0

echo %@isfloat[1234.5]
1
```

**4.4.5.153 @ISLOWER**

**@ISLOWER[*string*]** - returns 1 if *string* is composed only of lower case letters.

**Example:**

```
echo %@islower[hello]
1

echo %@islower[Hello]
0
```

**4.4.5.154 @ISODOWI**

**@ISODOWI[*date*]** : Returns the ISO 8601 numeric day of the week (Monday=1, Sunday=7).

**Example:**

```
echo %@isodowi[%_date]
7

echo %@isodowi[2012-01-02]
1
```

**4.4.5.155 @ISOWEEK**

**@ISOWEEK[*date*]** : Returns the ISO8601 numeric week of year.

**Example:**

```
echo %@isoweek[%_date]
23

echo %@isoweek[2012-23-02]
1
```

**4.4.5.156 @ISOWYEAR**

**@ISOWYEAR[*date*]** : Returns the ISO8601 numeric week date year.

**Example:**

```
echo %@isowyear[%_date]
2012
```

**4.4.5.157 @ISPRIME**

**@ISPRIME[*n*]** : Returns 1 if the (64-bit integer) *n* is a prime number.

**Example:**

```
echo %@isprime[7]
1

echo %@isprime[22]
0
```

```
echo %@isprime[30181]
1
```

#### 4.4.5.158 @ISPRINT

**@ISPRINT[*string*]**: Returns **1** if *string* is entirely composed of printable characters; **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISASCII](#), [@ISCNTRL](#), [@ISDIGIT](#), [@ISPUNCT](#), [@ISSPACE](#), [@ISXDIGIT](#).

**Examples:**

```
echo %@isprint[abc]
1
```

```
set var=abc^ndef
echo %@isprint[%var]
0
```

#### 4.4.5.159 @ISPROC

**@ISPROC[*pid*]** : Returns 1 if the specified process ID is an active process, or 0 if it is not.

**Example:**

```
echo %@pid[tcc.exe]
447988
```

```
echo %@isproc[447988]
1
```

#### 4.4.5.160 @ISPUNCT

**@ISPUNCT[*string*]**: Returns **1** if *string* is entirely composed of punctuation characters, i.e. printable characters which are not alphanumeric or space; **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISASCII](#), [@ISCNTRL](#), [@ISDIGIT](#), [@ISPRINT](#), [@ISSPACE](#), [@ISXDIGIT](#).

**Examples:**

```
echo %@ispunct[.]
1
```

```
echo %@ispunct[+]
1
```

```
echo %@ispunct[: -]
1
```

```
echo %@ispunct[.,a]
```

0

#### 4.4.5.161 @ISSPACE

**@ISSPACE[*string*]**: Returns **1** if *string* is entirely composed of white space characters (0x09 - 0x0D or 0x20); **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISASCII](#), [@ISCNTRL](#), [@ISDIGIT](#), [@ISPRINT](#), [@ISPUNCT](#), [@ISXDIGIT](#).

**Example:**

```
echo %@isspace[]
1

echo %@isspace[hello world]
0
```

#### 4.4.5.162 @ISUPPER

**@ISUPPER[*string*]** : Returns 1 if *string* is composed only of upper case letters.

**Example:**

```
echo %@isupper[HELLO]
1

echo %@isupper[Hello]
0
```

#### 4.4.5.163 @ISXDIGIT

**@ISXDIGIT[*string*]**: Returns **1** if *string* is entirely composed of hexadecimal digits (0- 9,A-F, a-f); **0** otherwise.

See also: [@ISALNUM](#), [@ISALPHA](#), [@ISASCII](#), [@ISCNTRL](#), [@ISDIGIT](#), [@ISPRINT](#), [@ISPUNCT](#), [@ISSPACE](#).

**Example:**

```
echo %@isxdigit[0]
0

echo %@isxdigit[7F]
1

echo %@isxdigit[0x7F]
1
```

#### 4.4.5.164 @JUNCTION

**@JUNCTION[*dir*]** : Returns the directory referenced by the specified junction.

#### 4.4.5.165 @LABEL

**@LABEL[*d:*]**: Returns the volume label of the specified disk drive. The drive letter must be followed by a colon.

**Examples:**

```
echo %@label[C:]
Windows7
```

```
echo %@label[%_disk:]
Development
```

See also: [VOL](#).

#### 4.4.5.166 @LCS

**@LCS[*string1, string2*]** : Returns a pointer to the Longest Common Subsequence in ***string1*** and ***string2***.

#### 4.4.5.167 @LEFT

**@LEFT[*n, string*]** : If ***n*** is positive, it returns the leftmost ***n*** characters of ***string***. If ***n*** is greater than the length of ***string***, it returns the entire ***string***. If ***n*** is negative, it returns ***string*** after dropping its rightmost ***n*** characters, unless **-n** is greater than the length of ***string***, in which case it returns an empty string.

**Examples:**

```
echo %@LEFT[2,jpsoft]
jp

echo %@LEFT[22,jpsoft]
jpsoft

echo %@LEFT[-2,jpsoft]
jps

echo "%@LEFT[-22,jpsoft]"
""
```

#### 4.4.5.168 @LEN

**@LEN[*string*]** : Returns the length of ***string***.

**Examples:**

```
echo %@len[this is a test]
14
```

```
echo %@len[%comspec]
41
```

#### 4.4.5.169 @LFN

**@LFN[filename]**: Returns the long filename for a short ("8.3") **filename**. The **filename** may contain any valid filename element including drive letter, path, filename and extension; the entire name including all intermediate paths will be returned in long name format. If **filename** does not refer to an actual file, the results are unpredictable.

On an LFN drive, the returned name may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. See the notes under [Variable Functions](#) for additional details.

**Example:**

```
echo "%@lfn[c:\progra~1]"
"C:\Program Files"
```

#### 4.4.5.170 @LINE

**@LINE[filename,n]**: Returns line **n** from the specified file. The first line in the file is numbered 0. **\*\*EOF\*\*** is returned for all line numbers beyond the end of the file.

The **filename** must be in quotes if it contains white space or special characters.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

@LINE works with files having lines of no more than 32,767 characters. Longer lines will not be counted accurately.

The @LINE function must read each line of the file to find the line you request, and will therefore cause significant delays if used in a long loop or on a large file. For a more effective method of processing each line of a file in sequence use the [DO](#) command, or [@FILEOPEN](#) and a sequence of [@FILEReads](#).

You can retrieve input from standard input if you specify **CON** as the filename. If you are [redirecting](#) input to @LINE using this feature, you must use [command grouping](#) or the redirection will not work properly (you can [pipe](#) to @LINE without a command group; this restriction applies only to input redirection). For example:

```
(echo %@line[con,0]) < myfile.dat
```

@LINE can retrieve data from file streams on NTFS drives if the stream name is specified. See [NTFS File Streams](#) for additional details on file streams.

Beware of characters with special meaning to **TCC**, such as redirection and piping symbols, within the file. Use [SETDOS](#) /X with appropriate codes as needed.

#### 4.4.5.171 @LINES

**@LINES[filename]**: Returns the line number of the last line in the file, or "-1" if the file is empty. The first line in the file is numbered 0, so (for example) @LINES will return 0 for a file containing one line. To get the actual number of lines, use %@INC[%@LINES[filename]].

The **filename** must be in quotes if it contains white space or special characters.

@LINES works with files having lines of no more than 32,767 characters; longer lines will not be counted accurately. @LINES must read each line of the file in order to count it, and will therefore cause significant delays if used on a large file.

@LINES can count lines in file streams on NTFS drives if the stream name is specified. See [NTFS File Streams](#) for additional details on file streams.

@LINES also sets two [command variables](#):

|                |                                                |
|----------------|------------------------------------------------|
| %_LINES_MAXLEN | The length of the longest line (in characters) |
| %_LINES_MAXLOC | The line number (base 0) of the longest line   |

**Example:**

```
echo %@lines[readme.txt]
170
```

#### 4.4.5.172 @LINKS

**@LINKS[filename]** : Returns the number of hard links for the specified file (NTFS only).

@LINKS may not work for remote files (depending on your network redirector and the type of server you are querying).

See also [MKLNK](#).

**Example:**

```
echo %@links[c:\windows\explorer.exe]
2
```

#### 4.4.5.173 @LOWER

**@LOWER[string]** : Returns the **string** converted to lower case.

**Examples:**

```
echo %@lower[This iSS aTeSt]
this is a test
```

```
echo %@lower[%path%]
c:\windows\system32
```

#### 4.4.5.174 @LTrim

**@LTrim[*string1*,*string2*]** : - Returns *string2* with all the leading characters in *string1* removed. *String1* must be enclosed in double quotes if it contains any spaces, tabs, or commas.

**Examples:**

```
echo "%@ltrim[JP,JP Software]"
 " Software"
```

#### 4.4.5.175 @Lua

**@Lua[*expression*]** : - Execute a Lua expression.

**Examples:**

```
echo %@lua[print 'foo']
 foo
```

#### 4.4.5.176 @MACAddress

**@MACAddress[*IPaddress*]**: Returns the unique Media Access Control (MAC) address of the network interface at *IPAddress*.. An invalid or unknown address will return an error (see [@ERRTEXT](#) to decipher the error number if necessary).

See also [@IPAddress](#).

**Example:**

```
echo %@macaddress[192.168.1.2]
 00-7e-18-d5-2d-09
```

#### 4.4.5.177 @MAKEAGE

**@MAKEAGE[*date*[,*time*[,*format*]]]** : Converts *date* and *time* (if included) to an age, a single value in the same format as [@FILEAGE](#).

@MAKEAGE accepts an optional third parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 (yyyy-Www-d)

## 6 ISO 8601 (yyyy-ddd)

`@MAKEAGE` can be used to compare the time stamp of a file with a specific date and time, for example:

```
if %@fileage[myfile] lt %@makeage[1/1/85] echo OLD!
```

[@AGEDATE](#) is the inverse of this function.

### Examples:

```
echo %@makeage[%_date]
12980736000000000000

echo %@makeage[%_date,%_time]
129808104040000000
```

See also: [Time Stamps](#), [@FILEAGE](#), [@AGEDATE](#).

### 4.4.5.178 @MAKEDATE

**`@MAKEDATE[n,d]`:** Returns a date, formatted according to the current country settings, or as explicitly specified by *d* (see [Date Display Formats](#)). *n* is interpreted as the number of days since 1980-01-01, and must be in the range 0 to 43829 (corresponding to the date 2099-12-31). This function is the inverse of [@DATE](#). The optional second parameter specifies the date format:

- |          |                       |
|----------|-----------------------|
| <b>0</b> | system default        |
| <b>1</b> | USA (mm/dd/yy)        |
| <b>2</b> | European (dd/mm/yy)   |
| <b>3</b> | Japan (yy/mm/dd)      |
| <b>4</b> | ISO 8601 (yyyy-mm-dd) |
| <b>5</b> | ISO 8601 (yyyy-Www-d) |
| <b>6</b> | ISO 8601 (yyyy-ddd)   |

### Examples:

```
echo %@makedate[7924]
09/11/01

echo %@makedate[7924,4]
2001-09-11
```

### 4.4.5.179 @MAKETIME

**`@MAKETIME[n]` :** Returns a time (formatted using the Time Separator specified in Regional Settings). *n* is interpreted as the number of seconds since midnight, and must not exceed 86399. This function is the inverse of [@TIME](#).

### Examples:

```
echo %@maketime[45240]
12:34:00
```

```
echo %@maketime[79244]
22:00:44
```

#### 4.4.5.180 @MAX

**@MAX[a,b,c,...]**: Returns the largest in the list of parameters. All parameters must be integers in the range **-2147483647** to **2147483647** and must be separated either by whitespace or by commas.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

**Example:**

```
echo %@max[1,5,2,0,-1]
5
```

#### 4.4.5.181 @MD5

String mode:   **@MD5[s,string]**  
File mode:   **@MD5[f,filename]**

Returns the 32 hexadecimal digit MD5 hash of the character in **string** or of the contents of the file **filename**. The first parameter must be **s** for a string, and any leading or trailing whitespace characters in **string** are included.

**Filename** may be specified with or without an optional **f**. @MD5 returns **-1** if the file does not exist, or it cannot be read.

Since **Take Command** handles all internal strings as Unicode, @MD5 will return different results for a string and the identical string in an ASCII file.

See also: [@SHA256](#), [@SHA384](#), [@SHA512](#), and [@CRC32](#).

**Example:**

```
echo %@md5[s,this is a string]
93D64091ADF43E8FC0B74257AFD82FC3
```

#### 4.4.5.182 @MIN

**@MIN[a,b,c,...]** : Returns the smallest in the list of parameters. All parameters must be integers in the range **-2147483647** to **2147483647** and must be separated either by whitespace or by commas.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

**Example:**

```
echo %@min[1,5,2,0,-1]
-1
```

#### 4.4.5.183 @MONTH

**@MONTH[date[,format]]** : Returns the month number for the specified date (1-12). See [date formats](#) for information on acceptable date formats.

**@MONTH** accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Examples:**

```
echo %@month[2010-01-01]
1

echo %@month[%_date]
5
```

#### 4.4.5.184 @MX

**@MX[address]** : Returns the email server for the specified user address.

**Example:**

```
echo %@mx[support@jpsoft.com]
direct-connect.jpsoft.com
```

#### 4.4.5.185 @NAME

**@NAME[filename]**: Returns the base name of a file, without the path or extension.

The **filename** must be in quotes if it contains white space or special characters. On an LFN drive, the returned filename may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. See the notes under [Variable Functions](#) for additional details.

**Note:** The @NAME function makes no assumption about the existence of a file or directory. Its **filename** parameter can be any string and the function will attempt to extract from it a base name.

**Examples:**

```
echo %@name[xyz.abc]
xyz

echo "%@name[%_comspec]"
"tcc"
```

#### 4.4.5.186 @NUMERIC

**@NUMERIC[*[+|-]string*]** : Returns **1** if **string** is numeric, and **0** otherwise.

To be numeric, the following must be true:

1. The first character may be a + or - sign,
2. The next character must be a decimal digit (0 to 9) or the [decimal separator](#).
3. The remainder of **string** must be composed entirely of decimal digits (0 to 9), the [thousands separator](#), and no more than a single decimal separator, with no thousands separators following the decimal separator.

Examples:

| <b>function</b>                     | <b>value</b> |
|-------------------------------------|--------------|
| <code>%@numeric[12345]</code>       | 1            |
| <code>%@numeric[-12345]</code>      | 1            |
| <code>%@numeric[.12345]</code>      | 1            |
| <code>%@numeric[\$12.34]</code>     | 0            |
| <code>%@numeric[5.00.125]</code>    | 0            |
| <code>%@numeric[+5.00.125,5]</code> | 0            |
| <code>%@numeric[.00.125]</code>     | 0            |
| <code>%@numeric[-5,.00.125]</code>  | 0            |

#### 4.4.5.187 @OPTION

**@OPTION[*directive*]** : Returns the current value of the requested configuration option. All directives which can be altered dynamically are supported. If **directive** is not supported, an error is returned.

For configuration directives, the current value returned may not match that stored in the *.INI* file.

For color directives, the current value is returned as a single number (0-255) combining foreground and background specifications. See [Colors, Color Names & Codes](#) for details.

Examples:

```
echo %@option[passiveftp]
Yes

echo %@option[stdcolors]
0
```

#### 4.4.5.188 @OWNER

**@OWNER[filename]**: Returns the owner of the specified file (if any).

**Examples:**

```
echo %@owner[c:\windows\explorer.exe]
NT SERVICE\TrustedInstaller
```

```
echo %@owner[v.exe]
ASUS-PC\Rex
```

#### 4.4.5.189 @PATH

**@PATH[filename]**: Returns the path portion of **filename**, if present, including the drive letter and a trailing backslash but not including the base name or extension. If the **filename** parameter doesn't contain path information, you may expand it first with the [@FULL](#) function.

The **filename** must be in quotes if it contains white space or special characters. On an LFN or NTFS drive, the returned filename may contain white space or other special characters. To avoid problems which could be caused by these characters, quote the returned name before you pass it to other commands. See the notes under [Variable Functions](#) for additional details.

**Note:** The **@PATH** function makes no assumption about the existence of a file or directory. Its **filename** parameter can be any string, and the function will attempt to remove from it a "base name".

**Examples:**

```
echo "%@path["c:\program files\xyz.abc"]
"c:\program files\""

echo "%@path[xyz.abc]"
""
```

#### 4.4.5.190 @PERL

**@PERL[expression]** : Executes the specified Perl expression. **@PERL** requires PerlScript, the WSH COM interface to Perl. PerlScript is available with Active State Perl (free from [www.activestate.com](http://www.activestate.com)).

#### 4.4.5.191 @PID

**@PID[filename]** : Returns the PID for specified name (or 0 if no match). If you have multiple copies of the same executable running, **@PID** will return the first one it finds.

**Example:**

```
echo %@pid[tcc.exe]
22420
```

#### 4.4.5.192 @PIDCOMMAND

**@PIDCOMMAND[pid]** : Returns the startup command line for the specified process ID.

```
set pid=%@pid[tcc.exe]
```

```
echo %@pidcommand[%pid]
"C:\Program Files\JPSoft\TCMD14x64\TCC.EXE"
```

#### 4.4.5.193 @PING

**@PING[host[,timeout[,packetsize]]]** : Returns the response time in milliseconds for the specified host. **Host** is the IP address or name, **timeout** is the maximum number of seconds to wait, and **packetsize** is the size of the data packet sent to the host in the ping request. The **timeout** defaults to 60 seconds, and **packetsize** defaults to 64 bytes. The minimum packet size is 12 bytes, and the maximum is 65520 bytes.

A negative value indicates an error. If the request times out, @PING returns -1. An unreachable host returns -2. An invalid address returns -3.

##### Examples:

```
echo %@ping[microsoft.com]
echo %@ping[microsoft.com,10]
echo %@ping[microsoft.com,,16]
echo %@ping[192.168.1.100,2,512]
```

#### 4.4.5.194 @PLUGINVER

**@PLUGINVER[plugin]** : Returns the version number (major.minor.build) for the specified plugin.

#### 4.4.5.195 @PPID

**@PPID[name]** : Returns the PID for the parent process of the specified name (or 0 if no match). If you have multiple copies of the same executable running, @PPID will return the parent PID for the first one it finds.

If the *name* argument begins with a =, it is assumed to be a PID instead of a process name.

##### Example:

```
echo %@ppid[tcc.exe]
21960
```

#### 4.4.5.196 @PRIME

**@PRIME[n]** : Returns the first prime number  $\geq n$  (a 64-bit integer).

##### Example:

```
echo %@prime[13798225]
13798247
```

#### 4.4.5.197 @PRIORITY

**@PRIORITY[pid]** : Returns the priority class for the specified process ID. The return values are (in hex):

|      |              |
|------|--------------|
| 8000 | Above normal |
| 4000 | Below normal |

```

100 Realtime
80 High
40 Idle
20 Normal

```

**Example:**

```

echo %@priority[33900]
20

```

**4.4.5.198 @PROCESSTIME**

**@PROCESSTIME[pid,n]** : Return the process time as a fileage. *n* is the time to return:

- 0 - Start time
- 1 - End time
- 2 - Kernel mode time
- 3 - User mode time

**Example:**

```

echo %@processtime[33900]
129811263230521496

```

**4.4.5.199 @PYTHON**

**@PYTHON[expression]** : Executes the specified Python expression.

The Python interpreter in **TCC** is persistent, so if you want to reset it pass an empty string to **@PYTHON**.

**4.4.5.200 @QUOTE**

**@QUOTE[string]** : Returns a double quoted argument if it contains any whitespace characters.

**Examples:**

```

echo %@quote[Now is the time]
"Now is the time"

echo %@quote[Now is the time]
Now is the time

```

**4.4.5.201 @RANDOM**

**@RANDOM[min, max]**: Returns a pseudo random integer value between *min* and *max*, inclusive. The random number generator is initialized from the system clock the first time it is used after **TCC** starts and will therefore produce a different sequence of numbers each time you use it. *Min* and *max* are 32-bit signed integers, so the allowable range is -2,147,483,647 to 2,147,483,647.

**Examples:**

```

echo %@random[0,1]
0

echo %@random[-10,10]

```

```
7

echo %@random[-10,10]
9

echo %@random[-10,10]
-6
```

#### 4.4.5.202 @READSCR

**@READSCR[row,col,length]**: Returns the text displayed in the **TCC** window at the specified location. The upper left corner of the window is location 0,0. The **row** and **column** can be specified as an offset from the current cursor location by preceding either value with a [+] or [-]. For example:

```
%@readscr [-2,+2,10]
```

returns 10 characters from the screen, starting 2 rows above and 2 columns to the right of the current cursor position.

#### 4.4.5.203 @READY

**@READY[d:]**: Returns **1** if the specified drive is ready; otherwise returns **0**. The drive letter must be followed by a colon.

**@READY** does not support UNC names.

#### Examples:

```
echo %@ready [E:]
0

echo %@ready [%_boot:]
1
```

#### 4.4.5.204 @REGBREAD

**@REGBREAD[HKEY...\\subkey\\value,handle,length]**: Read a value from the registry to a binary buffer.

*handle* : A buffer previously created with [@ALLOC](#).

*length* : The length (in bytes) to write to the registry key.

If **@REGBREAD** succeeds, it returns "0", otherwise it returns the Windows error number.

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

**Note:** Remember to use quotes around any entry containing spaces or commas!

See [@REGBWRITE](#) and [@ALLOC](#).

#### 4.4.5.205 @REGBWRITE

**@REGBWRITE[HKEY...\\subkey\\value,type,handle,length]:** Write a value from a binary buffer to the registry.

*type* : The type of key. @REGBWRITE supports keys of type REG\_BINARY and REG\_NONE.

*handle* : A buffer previously created with [@ALLOC](#).

*length* : The length (in bytes) to write to the registry key.

If @REGBWRITE succeeds, it returns "0", otherwise it returns the Windows error number.

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

**Note:** Remember to use quotes around any entry containing spaces or commas!

See [@REGREAD](#) and [@ALLOC](#).

#### 4.4.5.206 @REGCOPYKEY

**@REGCOPYKEY[HKEY...\\key, targetkey]** : Recursively copies the specified key and all of its subkeys to the target key. Returns 1 if the key was copied, 0 otherwise. The key names must be enclosed in double quotes if they contain any separator characters (space, comma, or tab).

Both keys must be in the same root (HKCR, HKCU, HKLM, HKU, or HKCC).

If you are running a 32-bit version of TCC in a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

**Note:** If you are copying a very large tree, this function can take several minutes to finish. (This is a Windows issue, not **TCC**.)

See [@REGCREATE](#) for information on the format of the key name.

#### 4.4.5.207 @REGCREATE

**@REGCREATE[HKEY...\\subkey]:** Create a new registry subkey. The parameter starts with the root key, which can be abbreviated:

| Full root key       | Short |
|---------------------|-------|
| HKEY_CLASSES_ROOT   | HKCR  |
| HKEY_CURRENT_USER   | HKCU  |
| HKEY_LOCAL_MACHINE  | HKLM  |
| HKEY_USERS          | HKU   |
| HKEY_CURRENT_CONFIG | HKCC  |

The remainder of the parameter (after the backslash) specifies the new subkey. The entire name must be quoted if it contains any white space or special characters, for example:

```
@REGCREATE ["HKLM\\Software\\My Company\\My Product\\User"]
```

REGCREATE will create any intermediate keys necessary. For example, @REGCREATE [ HKCU\\key1 \\key2\\key3] will create all three keys (if they do not already exist). REGCREATE returns 0 if the subkey was created or the Windows error number if an error occurred.

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name. (Not available in **TCC/LE**.) For example:

```
@regcreate["HKLM_64\Software\Company\Product\User"]
```

See also: [@REGQUERY](#) (read a value), [@REGSET](#) (write a value), and [@REGSETENV](#) (write and broadcast a value).

#### 4.4.5.208 @REGDELKEY

**@REGDELKEY[HKEY...\\key]** : Deletes the specified key and all of its subkeys. Returns **1** if the key was deleted, **0** otherwise. The expression must be enclosed in double quotes if it contains any separator characters (space, comma, or tab).

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

**Note:** use EXTREME caution with this function. It has the potential for causing irreparable damage to your registry and can even prevent Windows from booting!

See [@REGCREATE](#) for information on the format of the key name.

#### 4.4.5.209 @REGEX

**@REGEX[expression,string]** : Returns **1** if the **expression** was found and **0** if it was not. The expression must be enclosed in double quotes if it contains any separator characters (space, comma, or tab). See [Regular Expression Syntax](#) for supported expressions.

#### 4.4.5.210 @REGEXINDEX

**@REGEXINDEX[expression,string]** : Returns the offset of the first match. The **expression** must be enclosed in double quotes if it contains any separator characters (space, comma, or tab). See [Regular Expression Syntax](#) for supported expressions. (This function is basically a wildcard-enabled @INDEX.)

#### 4.4.5.211 @REGEXIST

**@REGEXIST[HKEY...\\key]** : Returns **1** if the specified key exists, **0** otherwise

The expression must be enclosed in double quotes if it contains any separator characters (space, comma, or tab).

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

See [@REGCREATE](#) for information on the format of the key name.

#### 4.4.5.212 @REGEXSUB

**@REGEXSUB[n,expression,string]** - returns the "nth" matching group in the string. (If you don't specify a group in **expression**, @REGEXSUB will return an empty string.) The expression must be enclosed in double quotes if it contains any separator characters (space, comma, or tab). See [Regular Expression Syntax](#) for supported expressions.

#### 4.4.5.213 @REGQUERY

**@REGQUERY[HKEY...\\subkey\\value]**: Read a value from the registry. REGQUERY supports keys of type REG\_DWORD, REG\_QWORD, REG\_EXPAND\_SZ, REG\_SZ, REG\_DWORD\_LITTLE\_ENDIAN, and REG\_QWORD\_LITTLE\_ENDIAN. If the key is of type REG\_EXPAND\_SZ, the value is returned without further expansion. If the value name does not exist, the function returns -1. If the value name is not supplied, REGQUERY returns the unnamed value for the specified key (the first value with a NULL name). To retrieve an unnamed value, add a trailing \ to the name.

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

**Note:** Remember to use quotes around any entry containing spaces or commas!

See [@REGCREATE](#) (create a subkey) for information on the format of the key name. See also: [@REGSET](#) (write a value) and [@REGSETENV](#) (write and broadcast a value).

#### 4.4.5.214 @REGSET

**@REGSET[HKEY...\\subkey\\value,type,data]**: Write a value to the registry. REGSET supports keys of type REG\_DWORD, REG\_SZ, REG\_EXPAND\_SZ, REG\_MULTI\_SZ, and REG\_DWORD\_LITTLE\_ENDIAN. **Type** is the value type (REG\_DWORD, REG\_EXPAND\_SZ, or REG\_SZ). **Data** is the data to set. If this parameter is not supplied, @REGSET will remove the value. REGSET returns 0 if the value was written or the Windows error number if an error occurred.

If you're setting REG\_MULTI\_SZ values, separate each *data* argument with a comma.

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

**Note:** Remember to use quotes around any entry containing spaces or commas!

See [@REGCREATE](#) for information on the format of the key name. See also: [@REGQUERY](#) (read a value) and [@REGSETENV](#) (write and broadcast a value).

#### 4.4.5.215 @REGSETENV

**@REGSETENV[HKEY...\\subkey\\value,type,data]** : The same as [@REGSET](#), but a broadcast message is sent to all applications when the change is made, so that any application monitoring such messages can respond to the change immediately if it is designed to do so. @REGSETENV returns 0 if the value was written or the Windows error number if an error occurred.

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

**Note:** Remember to use quotes around any entry containing spaces or commas!

See [@REGCREATE](#) for information on the format of the key name. See also: [@REGQUERY](#) (read a value) and [@REGSET](#) (write a value).

#### 4.4.5.216 @REGTYPE

**@REGTYPE[HKEY...\\key]** : Returns the registry variable type. The possible values are:

- 0 - REG\_NONE (No value type)
- 1 - REG\_SZ (Unicode null terminated string)
- 2 - REG\_EXPAND\_SZ (Unicode null terminated string with environment variable references)
- 3 - REG\_BINARY (Free form binary)
- 4 - REG\_DWORD (32-bit number)
- 5 - REG\_DWORD\_BIG\_ENDIAN (32-bit number)
- 6 - REG\_LINK (Symbolic Link)
- 7 - REG\_MULTI\_SZ (Multiple Unicode strings)
- 8 - REG\_RESOURCE\_LIST (Resource list in the resource map)
- 9 - REG\_FULL\_RESOURCE\_DESCRIPTOR (Resource list in the hardware description)
- 10 - REG\_RESOURCE\_REQUIREMENTS\_LIST
- 11 - REG\_QWORD (64-bit number)

If you are running a 64-bit version of Windows, you can access the 64-bit registry instead of the 32-bit registry by appending "\_64" to the HKEY name.

#### 4.4.5.217 @REMOTE

**@REMOTE[d:]**: Returns **1** if the specified drive is a remote (network) drive; otherwise returns **0**. The drive letter must be followed by a colon.

Examples:

```
echo %@remote[e:]
1

echo %@remote[%_disk]
0
```

#### 4.4.5.218 @REMOVABLE

**@REMOVABLE[d:]**: Returns **1** if the specified drive is removable (e.g. floppy disk, removable hard disk, USB storage device, etc.), **0** otherwise. The drive letter must be followed by a colon.

Examples:

```
echo %@removable[e:]
1

echo %@removable[%_disk]
0
```

#### 4.4.5.219 @REPEAT

**@REPEAT[char,count]** : Returns the character **char** repeated **count** times (**count** may not exceed 32,767).

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal

format ("0x" followed by a sequence of 0-F hex digits).

**Examples:**

| <b>function</b>         | <b>value</b> |
|-------------------------|--------------|
| %@repeat[%@char[95],10] |              |
| 7%@repeat[ ,7]spaces    | 7 spaces     |
| %@repeat[x,10]          | xxxxxxxxxx   |

#### 4.4.5.220 @REPLACE

**@REPLACE**[*string1, string2, text*]: Replaces all occurrences of **string1** in the **text** string with **string2**. For example, **%@replace[w,ch,warming]** returns the string "charming".

The search is case sensitive.

**Examples:**

```
echo %@replace[\,/, "ftp:\server\etc"]
"ftp://server/etc"

echo %@replace[^,,,A better, command processor]
A better command processor
```

#### 4.4.5.221 @REREPLACE

**@REREPLACE**[*source\_re,target\_re,source*] - Regular expression back reference replacement.

*source\_re* - Regular expression to apply to the source

*target\_re* - Regular expression for back reference

*source* - Source string

**Example:**

To replace the input string "a1.txt" with "1a.txt":

```
@REREPLACE[(.).).txt,\2\1.txt,a1.txt]
```

#### 4.4.5.222 @REVERSE

**@REVERSE**[*string*] : Reverses the order of the characters in **string**.

**Example:**

```
echo %@reverse[Now is the time for all good men]
nem doog lla rof emit eht si woN
```

**4.4.5.223 @REXX**

**@REXX[/=]expr**: Calls the REXX interpreter to execute the expression. Returns the numeric code or string result from REXX. Console output from the REXX interpreter is suppressed while executing the expression. Note that **TCC** expands variables and functions before passing **expr** to REXX.

**Examples:**

```
echo %@rexx[= 3 * 4]
set myprog=d:\path\xyz.exe
echo %@rexx[address(%@name[%myprog]); return address()]
```

**Note:** This function requires that an ooREXX (Object REXX) interpreter be installed and properly configured. See [REXX Support](#) for more information on the REXX language.

**4.4.5.224 @RIGHT**

**@RIGHT[n,string]** : If **n** is positive, it returns the rightmost **n** characters of **string**. If **n** is greater than the length of **string**, it returns the entire **string**. If **n** is negative, it returns **string** after dropping its leftmost **n** characters, unless **n** is greater than the length of **string**, in which case it returns an empty string.

**Examples:**

| function            | value        |
|---------------------|--------------|
| %@RIGHT[2,jpsoft]   | ft           |
| %@RIGHT[22,jpsoft]  | jpsoft       |
| %@RIGHT[-2,jpsoft]  | soft         |
| %@RIGHT[-22,jpsoft] | empty string |

**4.4.5.225 @RTRIM**

**@RTRIM[string1,string2]**: - Returns **string2** with any characters in **string1** removed from the right side of **string2**. **String1** must be enclosed in double quotes if it contains any spaces, tabs, or commas.

**Example:**

```
echo "%@rtrim[98XP,Windows XP]"
"Windows "
```

**4.4.5.226 @RUBY**

**@RUBY[expression]** : Returns the string result of the Ruby expression. Note that the Ruby environment is persistent within a **TCC** tab window, so you can do things like:

```
%@ruby[b = 42]
%@ruby[p b]
```

which will print "42". The value returned by **@RUBY** is the value returned by the RUBY API `rb_eval_string`.

You can query the type of the value returned by the last **@RUBY** call with the [RUBYTYPE](#) internal

variable, and the value returned by the last @RUBY call with the [RUBYVALUE](#) internal variable.

#### 4.4.5.227 @SCRIPT

**@SCRIPT[*engine,expression*]** : Returns the integer result of expression in the specified active scripting engine.

For example:

```
%@script[PerlScript,print "This message is from Perl!"]
```

See also the [SCRIPT](#) command.

#### 4.4.5.228 @SEARCH

**@SEARCH[*program[,path[,n]]*]** : Searches for **program** using the specified **path**, or, if not specified, the **PATH** environment variable, appending an extension if one isn't specified. (See [Executable Files and File Searches](#) for details on the default extensions used when searching **PATH**, the order in which the search proceeds, and the search of the **\WINDOWS** and **\WINDOWS\SYSTEM** directories.) Returns the fully expanded name of **program**, including drive, path, base name, and extension, or an empty string if a match is not found. If [wildcards](#) are used in the **program**, @SEARCH will search for the first program file that matches the wildcard specification, and returns the drive and path for that file plus the wildcard filename (e.g., *E:\UTIL\*.EXE*).

**Program** and each directory specification in **path** must be in quotes if they contain white space or special characters.

@SEARCH accepts an optional third parameter specifying whether to search the current directory. If **n** is 0, @SEARCH will not look for the file in the current directory. If **n** is 1 (the default), @SEARCH will look in the current directory before searching the path.

#### Examples:

```
echo %@search[notebook]
C:\Windows\system32\notebook.exe

echo %@search[msv*.dll,"d:\my dir\"]
D:\my dir\test\msvc.dll
```

#### 4.4.5.229 @SELECT

**@SELECT[*filename,top,left,bottom,right,title[,sort[,startline[,keymask]]]*]**: Pops up a selection window with the lines from the specified file, allowing you to display menus or other selection lists from within a batch file. You can move through the selection window with standard popup window navigation keystrokes, including string matching with wildcards or regular expressions (see [Popup Windows](#) for details; to change the navigation keys see [Key Mapping directives](#)).

**Filename** must be in quotes if it contains white space or special characters. The file size is limited only by available memory. To select from lines passed through input redirection or a pipe, use **CON: as filename**. To select from lines in the Windows clipboard, use **CLIP: as filename**.

If the specified width is < 150, the **top**, **left**, **bottom**, **right** parameters are assumed to be rows/columns relative to the TCC window. If the width is >= 150, the parameters are assumed to be screen coordinates

(in pixels).

If you set the optional 7th parameter *sorted* to 1, the list will be sorted alphabetically.

The optional 8th parameter *startline* specifies the line @SELECT should highlight at startup. (The first line is 1.) If you specify *startline*, you must also specify the *sort* parameter.

If you specify the optional 9th argument *keymask*, the searching is disabled, and TCC will check input keystrokes for a match against the key mask. If a match is found, @SELECT will return the current line and set the \_SELECT\_KEY environment variable to the input key value. The key mask is in the same format as INKEY /K.

The selected line number will be returned in the SELECT\_LINE environment variable (the first line is 1).

**Return value:**

- the text of the line the scrollbar is on if you press **Enter**
- an empty string if you press **Esc**.

**Example:**

```
call %@select["d:\path\my menu.txt",50,100,175,400>Select an option]
```

#### 4.4.5.230 @SERIAL

**@SERIAL[d:]**: Returns the serial number of the specified disk drive (in hex, i.e.: ABCD:0123). The drive letter must be followed by a colon.

**Examples:**

```
echo %@serial[C:]
1B:EB6D

echo %@serial[%_disk:]
F82B:746
```

See also: [@LABEL](#).

#### 4.4.5.231 @SERIALPORTCLOSE

**@SERIALPORTCLOSE[n]**: Close the serial port. *n* is the handle returned by a previous call to [@SERIALPORTOPEN](#).

#### 4.4.5.232 @SERIALPORTFLUSH

**@SERIALPORTFLUSH[n]**: Flush the contents of the serial port buffer. *n* is the handle returned by a previous call to [@SERIALPORTOPEN](#).

#### 4.4.5.233 @SERIALPORTOPEN

**@SERIALPORTOPEN[COMn[, baud[, parity[, bits[, flow]]]]]** - Open a serial port for read & write. The parameters are:

COMn - The COM port to open (COM1 - COM9)

**baud** - The baud rate (110 - 256000)

**parity** - The parity scheme to use. This can be one of the following values:

- no
- odd
- even
- mark
- space

**bits** - The number of bits in the bytes to transmit & receive

**flow** - The type of flow control to use. This can be one of the following values:

- no
- CtsRts
- CtsDtr
- DsrRts
- DsrDtr
- XonXoff

@SERIALPORTOPEN returns a handle to the serial port, which must be passed to the other serial port functions.

See also: [@SERIALPORTCLOSE](#), [@SERIALPORTFLUSH](#), [@SERIALPORTREAD](#), [@SERIALPORTWRITE](#).

#### 4.4.5.234 @SERIALPORTREAD

**@SERIALPORTREAD[n]:** Read the contents of the serial port buffer. *n* is the handle returned by a previous call to [@SERIALPORTOPEN](#).

#### 4.4.5.235 @SERIALPORTWRITE

**@SERIALPORTWRITE[n,text]:** Writes a string to the serial port. *n* is the handle returned by a previous call to [@SERIALPORTOPEN](#).

#### 4.4.5.236 @SERVER

**@SERVER[machinename,info]** : Returns information about the specified server ***machinename***, where ***info*** is the type of information you want. The types are:

**Name** - return the server name

**Comment** - return the server comment

**Version** - the OS version (major version + minor version).

**Users** - the number of users who can attempt to log on the server.

**Disconnect** - the auto-disconnect time, in minutes.

**Hidden** - returns 1 if the server is hidden, 0 if it is visible

**UserPath** - the path to user directories

**Type** - return the type of the server. This is a combination of the following hex flags (you can use the .AND. operator in IF / IFF to test individual flags):

|            |                                                              |
|------------|--------------------------------------------------------------|
| 1          | A LAN Manager workstation                                    |
| 2          | A LAN Manager server                                         |
| 4          | Any server running with Microsoft SQL Server                 |
| 8          | Primary domain controller                                    |
| 0x10       | Backup domain controller                                     |
| 0x20       | Server running the Timesource service                        |
| 0x40       | Apple File Protocol server                                   |
| 0x80       | Novell server                                                |
| 0x100      | LAN Manager 2.x domain member                                |
| 0x200      | Server sharing print queue                                   |
| 0x400      | Server running dial-in service                               |
| 0x800      | Unix/Linux server                                            |
| 0x1000     | Windows Server 2003, Windows XP, Windows 2000, or Windows NT |
| 0x2000     | Server running Windows for Workgroups                        |
| 0x4000     | Microsoft File and Print for NetWare                         |
| 0x8000     | Windows server that is not a domain controller               |
| 0x10000    | Server that can run the browser service                      |
| 0x20000    | Server running a browser service as backup                   |
| 0x40000    | Server running the master browser service                    |
| 0x80000    | Server running the domain master browser                     |
| 0x400000   | Windows 95/98/Me                                             |
| 0x1000000  | Server clusters available in the domain                      |
| 0x2000000  | Terminal Server                                              |
| 0x4000000  | Cluster virtual servers available in the domain              |
| 0x40000000 | Servers maintained by the browser                            |
| 0x80000000 | Primary domain                                               |

#### 4.4.5.237 @SERVICE

**@SERVICE[service,info]** : Returns information about the specified service.

**service** - the service name to query

**info** - the information you want

**1** The type of service. This will return one or more of the following values:

|     |                                                  |
|-----|--------------------------------------------------|
| 1   | Device driver                                    |
| 2   | File system driver                               |
| 16  | The service runs in its own process              |
| 32  | The service shares a process with other services |
| 256 | The service can interact with the desktop        |

**2** The current state of the service. This will return one of the following values:

|   |                            |
|---|----------------------------|
| 1 | The service is not running |
| 2 | The service is starting    |
| 3 | The service is stopping    |
| 4 | The service is running     |

- 5 The service continue is pending
- 6 The service pause is pending
- 7 The service is paused
  
- 3 Returns the check-point value the service increments to report its progress during a lengthy start, stop, pause, or continue operation. This value will be 0 if there is no pending operation.
  
- 4 The control codes the service accepts and processes in its handler function. This will return a combination of the following values (you can check the return value with the [@EVAL](#) OR test):
  - 1 The service can be stopped
  - 2 The service can be paused and continued
  - 4 The service is notified when system shutdown occurs
  - 8 The service can reread its startup parameters without being stopped and restarted
  - 16 The service is a network component that can accept changes in its binding without being stopped and restarted
  - 32 The service is notified when the computer's hardware profile has changed
  - 64 The service is notified when the computer's power status has changed
  - 128 The service is notified when the computer's session status has changed
  - 256 The service can perform pre-shutdown tasks
  
- 5 Returns the estimated time required for a pending start, stop, pause, or continue operation (in milliseconds).

#### 4.4.5.238 @SFN

**@SFN[filename]**: Returns the fully expanded short ("8.3") filename for a long **filename**. The **filename** may contain any valid filename element including drive letter, path, filename and extension. The entire name including all intermediate paths will be returned in short name format. If **filename** does not refer to an actual file, the results are unpredictable.

**Example:**

```
echo %@sfn[%comspec]
C:\PROGRA~1\JPSoft\TCMD13~1\TCC.EXE
```

#### 4.4.5.239 @SHA1

**@SHA1[filename]** : Returns the SHA1 checksum of the specified file.

**Example:**

```
echo %@sha1[c:\windows\notepad.exe]
7EB0139D2175739B3CCB0D1110067820BE6ABD29
```

See also [@SHA256](#), [@SHA384](#), [@SHA512](#), [@MD5](#), and [@CRC32](#).

#### 4.4.5.240 @SHA256

**@SHA256[filename]** : Returns the SHA2-256 checksum of the specified file.

**Example:**

```
echo %@sha256[c:\windows\notepad.exe]
142E1D688EF0568370C37187FD9F2351D7DDEDA574F8BFA9B0FA4EF42DB85AA2
```

See also [@SHA384](#), [@SHA512](#), [@MD5](#), and [@CRC32](#).

#### 4.4.5.241 @SHA384

**@SHA384[filename]** : Returns the SHA2-384 checksum of the specified file.

**Example:**

```
echo %@sha384[c:\windows\notepad.exe]
04BA669372BD3CBC40CAA9E44DE7C2760DBC27D68A79F7B0DC24048D6FF7A883CC2F0A6AB8
0AE6F4CD3E45045273873E
```

See also [@SHA256](#), [@SHA512](#), [@MD5](#), and [@CRC32](#).

#### 4.4.5.242 @SHA512

**@SHA512[filename]** : Returns the SHA2-512 checksum of the specified file.

**Example:**

```
echo %@sha512[c:\windows\notepad.exe]
2F37A2E503CFFBD7C05C7D8A125B55368CE11AAD5B62F17AAC7AAF3391A6886FA6A0FD732
23E9F30072419BF5762A8AF7958E805A52D788BA41F61EB084BFE8
```

See also [@SHA256](#), [@SHA384](#), [@MD5](#), and [@CRC32](#).

#### 4.4.5.243 @SHFOLDER

**@SHFOLDER[n]** : Returns the full pathname for the specified Windows folder (which vary in different versions of Windows and if the user has altered the defaults).

*n* is a number from 0 to 59 that returns the following values:

- 0 - Desktop
- 2 - Start Menu\Programs
- 5 - My Documents
- 6 - <user name>\Favorites
- 7 - Start Menu\Programs\Startup
- 8 - <user name>\Recent
- 9 - <user name>\SendTo
- 11 - <user name>\Start Menu
- 13 - "My Music" folder
- 14 - "My Videos" folder
- 16 - <user name>\Desktop
- 19 - <user name>\nethood
- 20 - windows\fonts
- 21 - templates
- 22 - All Users\Start Menu
- 23 - All Users\Start Menu\Programs
- 24 - All Users\Startup
- 25 - All Users\Desktop
- 26 - <user name>\Application Data

27 - <user name>\PrintHood  
28 - <user name>\Local Settings\Application Data (non roaming)  
29 - non localized startup  
30 - non localized common startup  
31 - common favorites  
32 - Internet cache  
33 - cookies  
34 - history  
35 - All Users\Application Data  
36 - Windows directory  
37 - Windows system directory  
38 - Program Files  
39 - <user name>\My Pictures  
40 - USERPROFILE  
41 - X86 system directory on x64  
42 - x86 c:\Program Files on x64  
43 - c:\Program Files\Common  
44 - x86 Program Files\Common on x64  
45 - All Users\Templates  
46 - All Users\Documents  
47 - All Users\Start Menu\Programs\Administrative Tools  
48 - <user name>\Start Menu\Programs\Administrative Tools  
53 - All Users\My Music  
54 - All Users\My Pictures  
55 - All Users\My Video  
56 - Resource Directory  
59 - USERPROFILE\Local Settings\Application Data\Microsoft\CD Burning

**Examples:**

**echo %@shfolder[42]**  
C:\Program Files (x86)

**echo %@shfolder[22]**  
C:\ProgramData\Microsoft\Windows\Start Menu

**4.4.5.244 @SIMILAR**

**@SIMILAR[string1,string2]** : Returns a value (0 - 100) reflecting the similarity between the two strings. **0** means the two strings have nothing in common; **100** means the strings are identical. Using the longer string as the first parameter usually results in lower similarity values and using the shorter results in higher values.

**Example:**

**echo %@similar[now is the time,then was the time]**  
75

**4.4.5.245 @SMCLOSE**

**@SMCLOSE[ handle ]** - Close a handle to shared memory.

**handle** - The handle returned by [@SMOPEN](#)

#### 4.4.5.246 @SMOPEN

**@SMOPEN[ size, name ]** - Open and return a handle to shared memory.

**size** - The size of shared memory (in bytes)

**name** - The name of the shared memory. The name can have a "Global\" or "Local\" prefix to create the object in the global or session namespace. If the name is "Global", and you are running Windows Vista or later, then only an elevated session can open the shared memory.

#### 4.4.5.247 @SMPEEK

**@SMPEEK[handle,offset,size]** : Read a value from shared memory.

**handle** - a handle from @SMOPEN

**offset** - the byte offset in the buffer (decimal or hex)

**size** - the size of the value to read (in bytes):

- 1 - character
- 2 - short
- 4 - int
- 8 - int64

#### 4.4.5.248 @SMPOKE

**@SMPOKE[handle,offset,size,value]** : Write a value to shared memory

**handle** - a handle from @SMOPEN

**offset** - the byte offset in the buffer (decimal or hex)

**size** - the size of the value (in bytes):

- 1 - character
- 2 - short
- 4 - int
- 8 - int64

**value** - the value to poke

@SMPOKE returns 0 on success.

#### 4.4.5.249 @SMREAD

**@SMREAD[ n,offset,type,length ]** - Read a string to shared memory

**n** - The shared memory handle returned by [@SMOPEN](#)

**offset** - The offset (in bytes) from the beginning of the shared memory buffer.

**type** - Either **a** to read the string as ASCII or **u** to read it as Unicode.

**length** - The length to read (in characters).

#### 4.4.5.250 @SMWRITE

**@SMWRITE[ *n,offset,type,string* ]** - Write a string to shared memory

***n*** - The shared memory handle returned by @SMOPEN

***offset*** - The offset (in bytes) from the beginning of the shared memory buffer.

***type*** - Either **a** to write the string as ASCII or **u** to write it as Unicode.

***string*** - The string to write.

#### 4.4.5.251 @SNAPSHOT

**@SNAPSHOT[DESKTOP | *window[,n]*]** : Save the desktop or a specific window to the clipboard as a BMP. The window argument can be either **DESKTOP** or a window title (which can include wildcards).

If the window argument is **DESKTOP**, the optional second argument specifies either which monitor (1 - **n**) whose desktop you want to save.

If the window argument is a window title, the optional second argument specifies whether you want the client area (0, the default) or the entire window (1) to be saved.

If the window argument begins with a =, it is assumed to be a PID instead of a window title.

@SNAPSHOT returns **0** if successful.

#### 4.4.5.252 @STRIP

**@STRIP[*chars,string*]** : Removes the characters in **chars** from the **string** and returns the result.

For example:

```
%@STRIP[AaEe,All Good Men]
```

returns "ll Good Mn".

The test is case sensitive.

To include a comma in the **chars** string, enclose the entire first parameter in quotes. @STRIP will remove the quotes before processing the **string**.

#### 4.4.5.253 @SUBSTR

**@SUBSTR[*string,start,length*]** : An older version of [@INSTR](#). If the **length** is omitted, it will default to the remainder of **string**. If **string** includes commas, it must be quoted with double quotes [""] or back-quotes [''], or each comma must be preceded by an [Escape character](#). The quotes count in calculating the position of the substring. @INSTR, which has **string** as its last parameter, does not have this restriction.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

#### Examples:

```
echo %@substr[this is useful,8]
useful

echo %@substr[this is useful,8,-2]
is

echo %@substr["commas, they DO matter",9]
they DO matter"

echo %@substr[commas^, they DO matter,9]
they DO matter
```

See also: [@INSTR](#).

#### 4.4.5.254 @SUBST

**@SUBST[n, string1, string2]**: Substitutes **string1** starting at position **n** in **string2**.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

#### 4.4.5.255 @SUMMARY

**@SUMMARY[file,property[,value]]** : Read or set NTFS SummaryInformation data for the specified file. If it is a compound file, @SUMMARY will retrieve the data from the compound file object; otherwise @SUMMARY will retrieve the data from the SummaryInformation stream attached to the file. The valid SummaryInformation fields are:

- Title
- Subject
- Author
- Keywords
- Comments
- Template
- LastAuthor
- Revision Number
- Edit Time
- Last printed
- Created
- Last Saved
- Page Count
- Word Count
- Char Count
- AppName

Note that most files won't have any of these fields; the ones that do will usually only have some, not all.

To set SummaryInformation data, specify the value in the optional third parameter.

For example, to set the **Title**:

```
@summary[foo.txt,Title,This is the Foo File]
```

#### 4.4.5.256 @SYMLINK

**@SYMLINK[link]** : Returns the target referenced by the specified symbolic link.

#### 4.4.5.257 @SYSTEMTIME

**@SYSTEMTIME[n]** : Return the system time as a [fileage](#). *n* is the time to return:

- 0 - The time that the system has been idle
- 1 - The time that the system has spent executing in Kernel mode (all threads in all processes, on all processors)
- 2 - The time that the system has spent executing in User mode (all threads in all processes, on all processors)

See also: [Time Stamps](#).

**Examples:**

```
echo %@systemtime[0]
39709212923676
```

```
echo %@systemtime[1]
39709212923676
```

```
echo %@systemtime[2]
39709212923676
```

#### 4.4.5.258 @TALNUM

**@TALNUM[string]**: Returns the number of alphabetic (a-z, A-Z) and/or numeric (0 - 9) characters in *string*.

See also: [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

**Example:**

```
echo %@talnum[123abc]
6

echo %@talnum[123 abc]
6

echo %@talnum[1-2-3]
3
```

#### 4.4.5.259 @TALPHA

**@TALPHA[string]**: Returns the number of alphabetic (a-z, A-Z) characters in *string*.

See also: [@TALNUM](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

**Example:**

```
echo %@talpha[abc123]
3

echo %@talpha[A B C]
3
```

**4.4.5.260 @TARCOUNT**

**@TARCOUNT[*tararchive*]:** Returns the number of files in a .tar archive.

See also [TAR](#) and [UNTAR](#).

**4.4.5.261 @TARCFILE**

**@TARCFILE[*tararchive,n*]:** Returns the compressed name of file *n* in a .tar archive.

See also [TAR](#) and [UNTAR](#).

**4.4.5.262 @TARDFILE**

**@TARDFILE[*tararchive,n*]:** Returns the decompressed name of file *n* in a .tar archive.

See also [TAR](#) and [UNTAR](#).

**4.4.5.263 @TARFILEDATE**

**@TARFILEDATE[*tararchive,n*]:** Returns the date and time of file *n* in a .tar archive.

See also [TAR](#) and [UNTAR](#).

**4.4.5.264 @TARFILESIZE**

**@TARFILESIZE[*tararchive,n*]:** Returns the size of file *n* in a .tar archive.

See also [TAR](#) and [UNTAR](#).

**4.4.5.265 @TASCII**

**@TASCII[*string*]:** Returns the number of 7-bit ASCII characters (0x00 - 0x7F) in *string*.

See also: [@TALNUM](#), [@TALPHA](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

**Examples:**

```
echo %@tascii[abc]
3

echo %@tascii[abc 123]
7
```

```
echo %@tascii["abc"]
5
```

#### 4.4.5.266 @TCL

**@TCL[*expression*]** : Returns the string result of the [Tcl](#) expression. (You cannot run a Tk script in @TCL, because there is no Tk event loop. If you want to run a Tk script, you need to execute it from the command line as you would a Tcl script, or with the @TK function.)

The Tcl interpreter in **TCC** is persistent, so if you want to reset it pass an empty string to @TCL.

See also [@TK](#).

#### 4.4.5.267 @TCNTRL

**@TCNTRL[*string*]**: Returns the number of ASCII control characters (0x00 - 0x1F or 0x7F) in *string*.

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

**Examples:**

```
echo %@tcntrl [abc]
0

set var=^r^n
echo %@tcntrl [%var]
2
```

#### 4.4.5.268 @TDIGIT

**@TDIGIT[*string*]** : Returns the number of digits (0-9) in *string*.

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

**Example:**

```
echo %@tdigit[0]
1

echo %@tdigit[123.456]
6

echo %@tdigit[-123]
3
```

#### 4.4.5.269 @TIME

**@TIME[*hh:mm:ss[am|pm]*]** : Returns the number of seconds since midnight for the specified time. We recommend that you use a 24-hour time format for compatibility with all locales. If "am" or "pm" are

specified @TIME will use a 12-hour format. Any non-numeric character, except a right bracket ] can be used to separate the hour, minute and second subfields.

#### Examples:

```
echo %@time[12:34:56]
45296
```

```
echo %@time[%_time]
81579
```

#### 4.4.5.270 @TIMER

**@TIMER[n[,precision]]** : Returns the current split time for a stopwatch started with the [TIMER](#) command. The value of **n** specifies the timer to read and can be 1, 2, or 3.

**@TIMER** accepts an optional second argument to return the timer split as a floating-point numeric value suitable for arithmetic. The possible values are:

- s** split time in seconds (2 digit decimal precision)
- m** split time in minutes (4 digit decimal precision)
- h** split time in hours (5 digit decimal precision)

#### 4.4.5.271 @TK

**@TK[expression]** : Returns the string result of the [Tk](#) expression. For example, this will execute the Tk script *test.tcl*:

```
echo %@tk[source test.tcl]
```

Because of the way the Tk interpreter works, it is not possible for **TCC** to maintain a persistent interpreter after executing a Tk script. **TCC** will close the current Tcl/tk interpreter and create a new one the next time @TCL is executed.

See also [@TCL](#).

#### 4.4.5.272 @TLOWER

**@TLOWER[string]** - returns the number of lower case letters in **string**.

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

#### Example:

```
echo %@tlower[hello]
5

echo %@tlower[Hello]
4
```

**4.4.5.273 @TPRINT**

**@TPRINT[*string*]**: Returns the number of printable characters in *string*.

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPUNCT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

**Examples:**

```
echo %@tprint[abc]
3
```

```
set var=abc^ndef
echo %@tprint[%var]
6
```

**4.4.5.274 @TPUNCT**

**@TPUNCT[*string*]**: Returns the number of punctuation characters in *string*, i.e. printable characters which are not alphanumeric or space.

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TSPACE](#), [@TUPPER](#), and [@TXDIGIT](#).

**Examples:**

```
echo %@tpunct[.]
1
```

```
echo %@tpunct[+]
1
```

```
echo %@tpunct[: -]
3
```

```
echo %@tpunct[.,a]
2
```

**4.4.5.275 @TRIM**

**@TRIM[*string*]** : Returns the string with the leading and trailing white space (space and tab characters) removed.

**Example:**

```
echo %@trim[this is a test string]
this is a test string
```

**4.4.5.276 @TRIMALL**

**@TRIMALL[*string*]** : Returns the string with the leading and trailing white space (space and tab characters), and any extra internal white space removed.

**Example:**

```
echo %@trimall[this is a test string]
 this is a test string
```

**4.4.5.277 @TRUENAME**

**@TRUENAME[filename]** : Returns the true, fully-expanded name for a file. @TRUENAME will "see through" junctions, symbolic links, a SUBST or network mapping. Wildcards cannot be used in the filename.

**Note:** The @TRUENAME function makes no assumption about the existence of a file or directory. Its **filename** parameter can be any string and the function will attempt to turn it into a fully qualified "volume + path + name" specification, whether that full reference exists or not.

**filename** must be in quotes if it contains white space or special characters.

**4.4.5.278 @TRUNCATE**

**@TRUNCATE[handle]** : Truncate the file opened for write access by [@FILEOPEN](#) at the current position of the file pointer, where **handle** is the value returned by [@FILEOPEN](#).

See also the related handle-based functions:

|                             |                                                |
|-----------------------------|------------------------------------------------|
| <a href="#">@FILECLOSE</a>  | Close a file handle                            |
| <a href="#">@FILEOPEN</a>   | Open a file handle                             |
| <a href="#">@FILEREAD</a>   | Read next line from a file handle              |
| <a href="#">@FILESEEK</a>   | Move a file handle pointer                     |
| <a href="#">@FILESEEKL</a>  | Move a file handle pointer to a specified line |
| <a href="#">@FILEWRITE</a>  | Write next line to a file handle               |
| <a href="#">@FILEWRITEB</a> | Write data to a file handle                    |

**4.4.5.279 @TSPACE**

**@TSPACE[string]**: Returns the number of white space characters (0x09 - 0x0D or 0x20) in **string**.

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TUPPER](#), and [@TXDIGIT](#).

**Example:**

```
echo %@tspace[]
 3

echo %@tspace[hello world]
 1
```

#### 4.4.5.280 @TUPPER

**@TUPPER[*string*]** - returns the number of upper case letters in *string*.

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), and [@TXDIGIT](#).

**Example:**

```
echo %@tupper[hello]
0

echo %@tupper[Hello]
1
```

#### 4.4.5.281 @TXDIGIT

**@TXDIGIT[*string*]**: Returns the number of characters in *string* that are hexadecimal digits (0-9 and A-F or a-f).

See also: [@TALNUM](#), [@TALPHA](#), [@TASCII](#), [@TCNTRL](#), [@TDIGIT](#), [@TLOWER](#), [@TPRINT](#), [@TPUNCT](#), [@TSPACE](#), and [@TUPPER](#).

**Example:**

```
echo %@txdigit[123abc]
6

echo %@txdigit[123 ttt]
3
```

#### 4.4.5.282 @UNC

**@UNC[*filename*]** : Returns the UNC name for the specified file (or an error if the file has no UNC, e.g., a local file).

#### 4.4.5.283 @UNICODE

**@UNICODE[*string*]** : Returns the space separated list of the Unicode values of the characters in *string*. You can use the [Escape character](#) before a special character (i.e., a quote or greater than (>) sign) in *string*.

See also: [@ASCII](#).

**Examples:**

| <b>function</b> | <b>value</b> |
|-----------------|--------------|
| %@unicode [a]   | 97           |
| %@unicode [A]   | 65           |
| %@unicode [^`]  | 96           |
| %@unicode [abc] | 97 98 99     |

#### 4.4.5.284 @UNIQUE

**@UNIQUE[*path*[,*prefix*]]** : Creates a zero-length file with a unique name in the specified directory, and returns its the full name and path. If no ***path*** is specified, the file will be created in the current directory. The file name will be FAT-compatible regardless of the type of drive on which the file is created. This function allows you to create a temporary file without overwriting an existing file.

The ***path*** must be in quotes if it contains white space or special characters.

If ***prefix*** is specified, @UNIQUE will use the first three characters as the first three characters of the unique filename.

Because the file is created, if the [Protect Redirected Output File](#) configuration option is set, you must use the style >! redirection to avoid errors.

Rapid, repeated, consecutive invocations of @UNIQUE may occasionally return a non-unique file name (the same name twice, for example), due to a long-standing timing bug in Windows. If you experience this problem you may need to use [DELAY](#), DELAY /M, or [BEEP](#) (with a frequency less than 20 Hz) to provide a short delay between invocations. You may also be able to work around the problem by performing some disk I/O activity between invocations, as this can force physical creation of the file on the disk before @UNIQUE is invoked again.

#### 4.4.5.285 @UNQUOTE

**@UNQUOTE[*string*]** : Returns the argument with all double quotes removed.

**Example:**

```
echo %@unquote["This is a """"heavily" quoted" string"]
This is a heavily quoted string
```

#### 4.4.5.286 @UNQUOTES

**@UNQUOTES[*string*]** : Returns the argument with leading and trailing double quotes removed.

**Example:**

```
echo %@unquotes["This is a """heavily" quoted" string"]
This is a ""heavily" quoted" string
```

**4.4.5.287 @UPPER**

**@UPPER[*string*]** : Returns *string* converted to upper case.

**Example:**

```
echo %@upper[this is a string]
THIS IS A STRING
```

**4.4.5.288 @URLDECODE**

**@URLDECODE[*string*]** : Decode an URL encoded string, replacing %xx with the original characters.

**4.4.5.289 @URLENCODE**

**@URLENCODE[*string*]** : Encode a string for Internet transmission, replacing non-alphanumeric characters with their %xx hex representation.

**4.4.5.290 @UTF8DECODE**

**@UTF8DECODE[*s, string*]** : Create a text string (using the current code page) from a UTF8 input string. Returns the text string.

**@UTF8DECODE[*infile, outfile*]** : Decode a UTF8 encoded file. Returns 0 if the output file was successfully written.

**4.4.5.291 @UTF8ENCODE**

**@UTF8ENCODE[*infile, outfile*]** : Encode a file from the current code page to UTF8. Returns 0 if the output file was successfully written.

**4.4.5.292 @UUDECODE**

**@UUDECODE[*infile, outfile*]** : Decode a UU encoded file. Returns 0 if the output file was successfully written.

**4.4.5.293 @UUENCODE**

**@UUENCODE[*infile, outfile*]** : Encode a UU encoded file. (3 bytes are encoded into 4 readable characters.) Returns 0 if the output file was successfully written.

**4.4.5.294 @UUID**

**@UUID[*n*]** : - Returns a UUID (same as a GUID in Windows). *n* can be:

- 0 - returns the UUID with lower case alphabetic characters and embedded hyphens
- 1 - returns the UUID with upper case alphabetic characters and embedded hyphens
- 2 - returns the UUID with lower case alphabetic characters and no hyphens
- 3 - returns the UUID with upper case alphabetic characters and no hyphens

**Examples:**

```
echo %@uuid[0]
2a1f64b4-d2cd-4e1b-accf-60effe6065f2
```

```
echo %@uuid[1]
94D8C597-5DD9-4947-95B5-B80B9EA223A0
```

```
echo %@uuid[2]
```

```
d9ccee3db2ad408fadea484cf8bdc977
```

```
echo %@uuid[3]
8A0EC71BD8ED4D1B986D071DF96426AE
```

#### 4.4.5.295 @VERINFO

**@VERINFO[filename[,info[,language]]]**: Returns the version information for the specified file. The optional second parameter specifies the desired information and defaults to **FileVersion**. The optional third parameter specifies the language/codepage pair (in hex). If that parameter is omitted, the code page for the default user language is assumed. If the requested information field is not provided in the specified file, returns a null string.

For example, **TCMD.EXE** returns values for:

```
CompanyName
FileDescription
FileVersion
InternalName
LegalCopyright
LegalTrademarks
OriginalFilename
ProductName
ProductVersion
Build
```

To return **CompanyName** :

```
echo %@verinfo[tcmd.exe,companyname,040904E4]
```

**Note:** Most, but not all, executables under Windows contain a **FileVersion** field. The number, names and contents of the specific information fields and language/codepage pairs provided within a given application can potentially be anything the programmer decided to use.

#### 4.4.5.296 @VERSION

**@VERSION[filename[,separator[,start[,force]]]]** : - Returns a serially "versioned" replacement for the file name. If the file doesn't exist, and *force* isn't set, @VERSION returns *filename*.

This is distinct from the function of **@UNIQUE[]** in that it retains the entire filename and only appends a version separator character and an ascending version number to the filename. @VERSION does not create the file; it just returns the next available version name.

@VERSION has four arguments:

- filename**      The filename to "versionize" (required)
- separator**     The version separation character (optional, defaults to ';'). Note that the [TCC include list](#) character is ;, so if you want to use ; in a filename, you will need to double quote the filename.
- start**        The starting version number (if necessary to add a version number; optional, defaults to '1')

**force**      The flag to force versioning, even if the file doesn't exist (optional, defaults to 0 or FALSE).

#### Examples:

```
echo %@version[myfile.txt]
myfile.txt;1

echo %@version[myfile.txt]
myfile.txt;2
```

#### 4.4.5.297 @WATTRIB

**@WATTRIB[filename[,-attributes[,p]]]**: If you do not specify any attributes, @WATTRIB returns the attributes of the specified file in the format **RHSADECIJNOPTV**, rather than **0** or **1**. If two or more parameters are specified, @WATTRIB returns a **1** if the specified file has the matching attribute(s); otherwise it returns a **0**. If the optional third argument ,**p** is included (partial match), then @WATTRIB will return **1** if any of the attributes match

This function is similar to [@ATTRIB](#), but supports file selection based on the following extended attributes available on NTFS volumes.

- E** Encrypted
- N** Normal
- T** Temporary
- P** Sparse file
- J** Junction or symbolic links
- L** Junction or symbolic links
- C** Compressed
- O** Offline
- I** Not content-indexed
- V** Virtualized

Attributes which are not set will be replaced with an underscore. For example, if **SECURE.DAT** has the read-only, hidden, and archive attributes set, **%@WATTRIB[SECURE.DAT]** would return **RH\_A\_\_\_\_\_**. If the file does not exist, @WATTRIB returns an empty string.

The attributes (other than **N**) can be combined (for example **%@ATTRIB[MYFILE,HS]**). For example, **%@WATTRIB[MYFILE,HS,p]** will return **1** if **MYFILE** has the hidden, system, or both attributes. Without ,**p** the function will return **1** only if **MYFILE** has both attributes (and no extended attributes).

**Filename** must be in quotes if it contains white space or special characters.

See also: [Attributes Switches](#) and the [ATTRIB](#) command.

#### 4.4.5.298 @WILD

**@WILD[string1,string2]** : Compares two strings and returns **1** if they match or **0** if they don't match. This function determines whether or not **string1** matches the pattern specified in **string2**, which may contain wildcards or [extended wildcards](#). No wildcards are permitted in **string1**. The test is not case

sensitive.

### Examples

The examples below assume that the **PATH** variable contains:

```
c:\windows;c:\windows\system32;"c:\program files\util";d:\jpsoft
```

| <b>string1</b> | <b>string2</b> | <b>match condition</b>                                                        | <b>result</b> |
|----------------|----------------|-------------------------------------------------------------------------------|---------------|
| %path          | * \UTIL*       | string \util anywhere                                                         | 1             |
| %path          | *c             | string ending with c                                                          | 0             |
| %path          | *t             | string ending with t                                                          | 1             |
| %path          | c*             | string starting with c                                                        | 1             |
| %path          | t*             | string starting with t                                                        | 0             |
| %path          | *c*            | string containing c                                                           | 1             |
| %path          | *t*            | string containing t                                                           | 1             |
| %path          | *b*            | string containing b                                                           | 0             |
| xyz            | ?              | one character long string                                                     | 0             |
| x              | ?              | one character long string                                                     | 1             |
| %path          | c?*            | leading c, followed by any one character, followed by 0 or more characters    | 1             |
| %path          | c*?            | leading c, followed by zero or more characters, followed by any one character | 1             |

#### 4.4.5.299 @WINAPI

**@WINAPI**[module,function[,integer | PINT=n | PLONG=n | PDWORD=n | NULL | BUFFER | "string"] :  
Returns the result of calling a Windows API function. The arguments are:

**module** - name of the DLL containing the function

**function** - function name (case sensitive)

**integer** - an integer value to pass to the function

**PINT** - a pointer to the integer *n*

**PLONG** - a pointer to the long integer *n*

**PDWORD** - a pointer to the DWORD *n*

**NULL** - a null pointer (0)

**BUFFER** - @WINAPI will pass an address for an internal buffer for the API to return a Unicode string value.

**aBUFFER** - @WINAPI will pass an address for an internal buffer for the API to return an ASCII string value.

**"string"** - text argument (this must be enclosed in double quotes). If the argument is preceded by an 'a' (i.e., a"Argument") then it is converted from Unicode to ASCII before calling the API. (Some Windows APIs only accept ASCII arguments.)

@WINAPI supports a maximum of 8 arguments. The return value is either a string value returned by the API (if BUFFER or aBUFFER is specified), or the integer value returned by the API. The function must be defined as WINAPI (`__stdcall`). If @WINAPI can't find the specified function, it will append a "W" (for the Unicode version) to the function name and try again.

See also [@CAPI](#).

#### 4.4.5.300 @WINCLASS

**@WINCLASS[*classname*]** : Returns the window title of the first window with the specified class name, or an empty string if no windows match.

#### 4.4.5.301 @WINCLIENTSIZE

**@WINCLIENTSIZE[*title*]** : Returns the client window size in the format *height, width*

If the *title* argument begins with a =, it is assumed to be a PID instead of a window title.

#### 4.4.5.302 @WINEXENAME

**@WINEXENAME[*title*]**: Returns the executable name for the first window matching *title* (which can include [wildcards](#)), or an empty string if none.

If the *title* argument begins with a =, it is assumed to be a PID instead of a window title.

#### 4.4.5.303 @WININFO

**@WININFO[*n*]**: Returns information about the current system. *n* is a number specifying what information to return:

| <i>n</i> | <i>Information returned</i>                                                   |
|----------|-------------------------------------------------------------------------------|
| 1        | Processor architecture<br>0 INTEL<br>6 IA64<br>9 x64 (AMD or Intel)           |
| 2        | Processor bit mask (set of configured processors)                             |
| 3        | Number of processors                                                          |
| 4        | Type of processor<br>586 Pentium:<br>2200 Intel IA64<br>8664 AMD or Intel x64 |
| 5        | Processor level                                                               |
| 6        | Processor revision                                                            |
| 7        | page size, bytes                                                              |
| 8        | virtual memory allocation granularity, bytes                                  |

#### 4.4.5.304 @WINMEMORY

**@WINMEMORY[*n*]** : Returns the requested Windows memory information. All values except memory load are returned in bytes. *n* is a number specifying what to return:

| <i>n</i> | <i>Information returned</i>               |
|----------|-------------------------------------------|
| <b>0</b> | Memory load, %                            |
| <b>1</b> | Total physical RAM                        |
| <b>2</b> | Available physical RAM                    |
| <b>3</b> | Total that can be stored in the page file |
| <b>4</b> | Available page file                       |
| <b>5</b> | Total virtual memory for process          |
| <b>6</b> | Total free virtual memory for process     |

#### 4.4.5.305 @WINMETRICS

**@WINMETRICS[n]** : Returns the requested Windows system metric. All screen dimension metrics are returned in pixels. *n* is a number determining which metric to return.

**Note:** This function provides direct access to the **GetSystemMetrics** API. Not all available parameters are listed here and your Windows configuration may support additional parameters. See your Windows technical documentation for details.

| <i>n</i>  | <i>Information returned</i>                                             |
|-----------|-------------------------------------------------------------------------|
| <b>0</b>  | Width of screen on primary display monitor                              |
| <b>1</b>  | Height of screen on primary display monitor                             |
| <b>2</b>  | Width of the vertical scroll bar                                        |
| <b>3</b>  | Height of the horizontal scroll bar                                     |
| <b>4</b>  | Height of title bar                                                     |
| <b>5</b>  | Width of window border                                                  |
| <b>6</b>  | Height of window border                                                 |
| <b>7</b>  | Width of dialog box border                                              |
| <b>8</b>  | Height of dialog box border                                             |
| <b>9</b>  | Height of thumb box on vertical scroll bar                              |
| <b>10</b> | Width of thumb box on horizontal scroll bar                             |
| <b>11</b> | Width of icon                                                           |
| <b>12</b> | Height of icon                                                          |
| <b>13</b> | Width of cursor                                                         |
| <b>14</b> | Height of cursor                                                        |
| <b>15</b> | Height of single line menu bar                                          |
| <b>16</b> | Width of client area for full-screen window on primary display monitor  |
| <b>17</b> | Height of client area for full-screen window on primary display monitor |
| <b>18</b> | Height of Kanji window                                                  |
| <b>19</b> | Mouse present flag<br>0 no<br>1 yes                                     |
| <b>20</b> | Height of arrow bitmap on vertical scroll bar                           |
| <b>21</b> | Width of arrow bitmap on horizontal scroll bar                          |
| <b>22</b> | Debug version of Windows<br>0 no<br>1 yes                               |
| <b>23</b> | Left and right mouse buttons swapped                                    |

|    |                                                                            |
|----|----------------------------------------------------------------------------|
|    | 0 no<br>1 yes                                                              |
| 28 | Minimum width of a window                                                  |
| 29 | Minimum height of a window                                                 |
| 30 | Width of bitmaps in title bar                                              |
| 31 | Height of bitmaps in title bar                                             |
| 32 | Width of window frame that can be sized                                    |
| 33 | Height of window frame that can be sized                                   |
| 34 | Minimum tracking width of window                                           |
| 35 | Minimum tracking height of window                                          |
| 41 | Is Pen Windows installed?<br>0 no<br>1 yes                                 |
| 42 | Is DBCS version of USER.EXE installed?<br>0 no<br>1 yes                    |
| 43 | Number of buttons on mouse                                                 |
| 70 | Windows will display visual info in place of audible info<br>0 no<br>1 yes |
| 73 | Computer has a slow processor<br>0 no<br>1 yes                             |
| 74 | Is Windows set up for Arabic/Hebrew?<br>0 no<br>1 yes                      |
| 75 | Mouse has a wheel<br>0 no<br>1 yes                                         |
| 76 | Coordinate of left side of virtual screen                                  |
| 77 | Coordinate of top of virtual screen                                        |
| 78 | Width in pixels of virtual screen                                          |
| 79 | Height in pixels of virtual screen                                         |
| 80 | Number of monitors on desktop                                              |

#### 4.4.5.306 @WINPID

**@WINPID[title]** : Returns the process ID for the window with the specified title, or -1 if no match is found.

**Example:**

```
echo %@winpid[TCC Prompt]
438636
```

#### 4.4.5.307 @WINPOS

**@WINPOS[title]**: Returns the screen coordinates of the window with the specified title, in the format "top,left,bottom,right".

If the *title* argument begins with a =, it is assumed to be a PID instead of a window title.

**Example:**

```
echo %@winpos[TCC Prompt]
25,25,367,702
```

**4.4.5.308 @WINSIZE**

**@WINSIZE[*title*]** : Returns the window size in the format *height, width*

If the *title* argument begins with a =, it is assumed to be a PID instead of a window title.

**Example:**

```
echo %@winsize[TCC Prompt]
342,677
```

**4.4.5.309 @WINSTATE**

**@WINSTATE[*title*]** : Returns the window state of the first window matching *title* (which can include [wildcards](#)). The return values are:

| Value | Window state |
|-------|--------------|
| 0     | Hidden       |
| 1     | Normal       |
| 2     | Minimized    |
| 3     | Maximized    |

If the *title* argument begins with a =, it is assumed to be a PID instead of a window title.

**4.4.5.310 @WINSYSTEM**

**@WINSYSTEM[*n*[,*v*]]**: Sets or returns the value of the requested Windows system-wide parameters.

To retrieve a parameter, the format is **%@winsystem[n]** where *n* is the appropriate **GET** number from the table below.

To set a parameter, the format is **%@winsystem[n,v]** where *n* is the appropriate **SET** number from the table below and *v* is the desired new value for that parameter.

Where the selection is a state, the legal values are **0** for off-disabled, and **1** for on/enabled.

Where the selection is a width or height, the values are in pixels.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

**Note:** This function provides direct access to the **SystemParametersInfo** API. Not all available parameters are listed here. See your Windows technical documentation for details, and use with caution.

|            |            |                                |
|------------|------------|--------------------------------|
| <b>GET</b> | <b>SET</b> | <b>Parameter to GET or SET</b> |
|------------|------------|--------------------------------|

|      |      |                                                                                                                          |
|------|------|--------------------------------------------------------------------------------------------------------------------------|
| 1    | 2    | Beep state                                                                                                               |
| 5    | 6    | Border width                                                                                                             |
| 10   | 11   | Keyboard repeat speed (0 to 31)                                                                                          |
| 13   | 13   | Width of an icon cell                                                                                                    |
| 14   | 15   | Screen saver time-out (seconds)                                                                                          |
| 16   | 17   | Screen saver state                                                                                                       |
| 22   | 23   | Keyboard repeat delay setting (0-3).                                                                                     |
| 24   | 24   | Height of an icon cell                                                                                                   |
| 25   | 26   | Icon title wrapping state                                                                                                |
| 27   | 28   | Pop-up menu alignment                                                                                                    |
| 37   | 38   | Full-window dragging state                                                                                               |
| 56   | 57   | Show Sounds accessibility flag                                                                                           |
| 68   | 69   | Keyboard preference state (0=mouse, 1=keyboard)                                                                          |
| 70   | 71   | Screen reviewer utility state                                                                                            |
| 74   | 75   | Font smoothing feature state                                                                                             |
| 79   | 81   | Time-out for the low-power phase of screen saving (seconds)                                                              |
| 80   | 82   | Time-out value for the power-off phase of screen saving (seconds)                                                        |
| 83   | 85   | Low-power phase of screen saving state                                                                                   |
| 84   | 86   | Power-off phase of screen saving state                                                                                   |
| 89   | 90   | Locale identifier for the system default input language.                                                                 |
| 93   | 94   | Mouse Trails feature state. (0 or 1= disabled, >1= number of cursors in the trail)                                       |
| 95   | 95   | Snap-to-default-button feature state                                                                                     |
| 98   | 99   | Width of the mouse pointer WM_MOUSEHOVER message trigger rectangle                                                       |
| 100  | 101  | Height of the mouse pointer WM_MOUSEHOVER message trigger rectangle                                                      |
| 102  | 103  | Time in the hover rectangle for the mouse pointer to trigger a WM_MOUSEHOVER message (milliseconds)                      |
| 104  | 105  | Number of lines to scroll when the mouse wheel is rotated                                                                |
| 106  | 107  | Time that the system waits before displaying a shortcut menu when the mouse cursor is over a submenu item (milliseconds) |
| 110  | 111  | IME status window state - per user (0=invisible, 1=visible)                                                              |
| 112  | 113  | Current mouse speed (1 to 20).                                                                                           |
| 4096 | 4097 | Active window tracking state                                                                                             |
| 4098 | 4099 | Menu animation feature state.                                                                                            |
| 4100 | 4101 | Combo box animation state.                                                                                               |
| 4102 | 4103 | List box smooth-scrolling effect state.                                                                                  |
| 4104 | 4105 | Gradient effect for window title bars.                                                                                   |
| 4106 | 4107 | Menu access keys underline state.                                                                                        |
| 4108 | 4109 | Active window tracking Z-order state.                                                                                    |
| 4110 | 4111 | Hot-tracking state.                                                                                                      |
| 4114 | 4115 | Menu fade animation state.                                                                                               |
| 4116 | 4117 | Selection fade effect state.                                                                                             |
| 4118 | 4119 | ToolTip animation state.                                                                                                 |
| 4120 | 4121 | Type of ToolTip animation (1 for fade, 0 for slide)                                                                      |
| 4122 | 4123 | Cursor shadow state.                                                                                                     |
| 4124 | 4125 | State of the Mouse Sonar feature                                                                                         |
| 4126 | 4127 | Mouse clicklock state                                                                                                    |
| 4128 | 4129 | Mouse vanish feature state                                                                                               |

|      |      |                                                                                                                                      |
|------|------|--------------------------------------------------------------------------------------------------------------------------------------|
| 4130 | 4131 | Whether native User menus have flat menu appearance.                                                                                 |
| 4132 | 4133 | Drop shadow effect state.                                                                                                            |
| 4158 | 4159 | State of all UI effects.                                                                                                             |
| 8192 | 8193 | Time following user input during which the system will not allow applications to force themselves into the foreground (milliseconds) |
| 8194 | 8195 | Active window tracking delay (milliseconds)                                                                                          |
| 8196 | 8197 | The number of times SetForegroundWindow will flash the taskbar button when rejecting a foreground switch request.                    |
| 8198 | 8199 | Caret width in edit controls                                                                                                         |
| 8200 | 8201 | Time delay before the primary mouse button is locked.                                                                                |
| 8202 | 8203 | Type of font smoothing (32769=standard anti-aliasing, 32770=ClearType).                                                              |
| 8204 | 8205 | Contrast value used in ClearType smoothing (1000-2200)                                                                               |
| 8206 | 8207 | Width of the left and right edges of the focus rectangle                                                                             |
| 8208 | 8209 | Height of the top and bottom edges of the focus rectangle                                                                            |

| GET | SET | Parameter to GET or SET |
|-----|-----|-------------------------|
|-----|-----|-------------------------|

#### 4.4.5.311 @WINTITLE

**@WINTITLE[pid]**: Returns the window title of the process with the specified process ID.

**Example:**

```
echo %@wintitle[15380]
TCC Prompt
```

#### 4.4.5.312 @WMI

**@WMI[namespace, "wql search"[,enum]]**: Returns the result of the WMI query.

The optional **enum** parameter specifies the property instance to return for classes that return multiple properties. You can omit the **enum** parameter if you're querying a single property and instance.

For details on what information is available, see the WMI and WQL documentation on MSDN ([msdn.microsoft.com](http://msdn.microsoft.com)).

See also [WMIQUERY](#).

**Examples:**

```
%@wmi[root\cimv2,"SELECT name FROM Win32_Processor"]
```

```
%@wmi[root\cimv2,"SELECT name, state FROM Win32_service",4]
```

#### 4.4.5.313 @WORD

**@WORD[["sep\_list",]n,string]** : Returns the *n*th word in *string*. The first (leftmost) word is numbered 0. If *n* is negative, words are counted backwards from the end of *string*, and the absolute value of *n* is used. You can specify the rightmost word by setting *n* to -0.

You can specify a range of words to return with the syntax:

**@WORD[["sep\_list",]start[-end | +range],string]**

Specify an inclusive range with a -. For example:

**%@word[2-4,A B C D E F G]** will return "C D E". (Note that you cannot use inclusive ranges when starting from the end.)

You can specify a relative range with a +. For example:

**%@word[2+1,A B C D E F G]** will return "C D".

The default list of separators for [@FIELD](#), [@FIELDS](#), [@WORD](#) and [@WORDS](#) consists of space, tab, and comma. You can use the optional first parameter, *sep\_list*, to specify the separators that you wish to use. If you want to use a quote mark as a separator, prefix it with an [Escape character](#). Alphabetic characters in *sep\_list* are case sensitive.

[@FIELD](#) and [@FIELDS](#) differ from [@WORD](#) and [@WORDS](#) in how multiple consecutive separators are counted. [@WORD](#) and [@WORDS](#) consider a sequence as a single separator, and ignore separators at either end of string. In contrast, [@FIELD](#) and [@FIELDS](#) count each occurrence of a separator individually, including those at either end of string.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits). To use hexadecimal form for a negative *n*, remember to use 32-bit 2's complement arithmetic, e.g., **0xFFFFFFFF** for -1.

See also: [@WORDS](#), [@FIELD](#), [@FIELDS](#).

**Examples:**

| function                              | value |
|---------------------------------------|-------|
| <b>%@WORD[2,NOW, , , IS THE TIME]</b> | THE   |
| <b>%@WORD[-0,NOW IS THE TIME]</b>     | TIME  |
| <b>%@WORD[-2,NOW IS THE TIME]</b>     | IS    |
| <b>%@WORD[ "=" ,1,2 + 2=4 ]</b>       | 4     |

#### 4.4.5.314 @WORDS

**@WORDS[["sep\_list",]string]**: Returns the number of words in *string*.

The default list of separators for [@FIELD](#), [@FIELDS](#), [@WORD](#) and [@WORDS](#) consists of space, tab, and comma. You can use the optional first parameter, *sep\_list*, to specify the separators that you wish to use. If you want to use a quote mark as a separator, prefix it with an [Escape character](#). Alphabetic characters in *sep\_list* are case sensitive.

[@FIELD](#) and [@FIELDS](#) differ from [@WORD](#) and [@WORDS](#) in how multiple consecutive separators are

counted. [@WORD](#) and [@WORDS](#) consider a sequence as a single separator, and ignore separators at either end of **string**. In contrast, [@FIELD](#) and [@FIELDS](#) count each occurrence of a separator individually, including those at either end of **string**.

If **string** is double quoted, you must specify **sep\_list**.

See also: [@WORD](#), [@FIELD](#), [@FIELDS](#).

#### 4.4.5.315 @WORKGROUP

**@WORKGROUP[*name*]**: Returns the workgroup of the computer specified by the DNS or NetBios **name**. If **name** is not specified, @WORKGROUP returns the workgroup of the local computer. (To query a remote computer, you must be an authenticated user on that computer.)

#### 4.4.5.316 @XMLCLOSE

**@XMLCLOSE[]** : Close an XML file previously opened by [@XMLOPEN](#).

For more details on XML support, see [XML in Take Command](#).

#### 4.4.5.317 @XMLNODES

**@XMLNODES[["filename"],path]** : Return the number of nodes (children) for the specified path in an XML file . The arguments are:

**filename** - name of XML file

**path** - one or more element accessors separated by a /.

If you don't specify a filename (which \*must\* be in double quotes), @XMLPATH will use the XML file previously opened by [@XMLOPEN](#).

For more details on XML support, see [XML in Take Command](#).

#### 4.4.5.318 @XMLOPEN

**@XMLOPEN[*filename*]** - open an XML file for use by [@XMLPATH](#) and/or [@XMLNODES](#).

For more details on XML support, see [XML in Take Command](#).

#### 4.4.5.319 @XMLPATH

**@XMLPATH[["filename"],path]** : XML XPath query. (See the [XML XPath docs](#) for details on XPath syntax.) The arguments are:

**filename** - name of XML file

**path** - one or more element accessors separated by a /.

If you don't specify a filename (which \*must\* be in double quotes), @XMLPATH will use the XML file previously opened by [@XMLOPEN](#).

To return an attribute, preface the attribute name with an @.

For more details on XML support, see [XML in Take Command](#).

#### 4.4.5.320 @YEAR

**@YEAR[date[,format]]**: Returns the year for the specified date. See [date formats](#) for valid formats.

@YEAR accepts an optional second parameter specifying the date format:

- 0 default
- 1 USA (mm/dd/yy)
- 2 Europe (dd/mm/yy)
- 3 Japan (yy/mm/dd)
- 4 ISO (yyyy-mm-dd)
- 5 ISO 8601 yyyy-Www-d
- 6 ISO 8601 yyyy-ddd

**Example:**

```
echo %@year[5-5-2012,1]
2012
```

#### 4.4.5.321 @YDECODE

**@YDECODE[s,string]** : Create a text string from a hexadecimal input string. Returns the text string.  
**@YDECODE[inputfile,outputfile]** : Decode a hex encoded file. Returns 0 if the output file was successfully written.

Y Encoding is similar to Base64, but uses 8-bit encoding to reduce the amount of data being sent and received.

#### 4.4.5.322 @YENCODE

**@YENCODE[s,string]** : Create a text string from a hexadecimal input string. Returns the text string.  
**@YENCODE[inputfile,outputfile]** : Encode a file. Returns 0 if the output file was successfully written.

Y Encoding is similar to Base64, but uses 8-bit encoding to reduce the amount of data being sent and received.

**Example:**

```
echo %@yencode[data.file,data.file.yenc]
```

#### 4.4.5.323 @ZIPCFILE

**@ZIPCFILE[ziparchive,n]**: Returns the compressed name of file *n* in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.324 @ZIPCFILESIZE

**@ZIPCFILESIZE[ziparchive,n]**: Returns the compressed size of file *n* in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.325 @ZIPCOMMENT

**@ZIPCOMMENT[ziparchive]**: Returns the comment (if any) for a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.326 @ZIPCOUNT

**@ZIPCOUNT[ziparchive]:** Returns the number of files in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.327 @ZIPFILE

**@ZIPFILE[ziparchive,n]:** Returns the decompressed name of file *n* in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.328 @ZIPFILECRC

**@ZIPFILESIZE[ziparchive,n]:** Returns the CRC of file *n* in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.329 @ZIPFILESIZE

**@ZIPFILESIZE[ziparchive,n]:** Returns the decompressed size of file *n* in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.330 @ZIPFILECOMMENT

**@ZIPFILECOMMENT[ziparchive,n]:** Returns the comment (description) of file *n* in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

#### 4.4.5.331 @ZIPFILEDATE

**@ZIPFILEDATE[ziparchive,n]:** Returns the date and time of file *n* in a .zip archive.

See also [ZIP](#) and [UNZIP](#).

## 4.5 Command Line

A **TCC** window displays a prompt when it is waiting for you to enter a command. The actual text depends on the current drive and directory as well as your [PROMPT](#) settings. (The default will look something like **[c:]>**). This is called the command line and the prompt is asking you to enter a command.

This section explains the features that will help you while you are entering commands, how keystrokes are interpreted when you enter them at the command line, and how to transfer text between **TCC** and other applications.

The keystrokes discussed here are the ones normally used by **TCC**. If you prefer using different keystrokes to perform these functions, you can assign new ones with [key mapping directives](#).

Some of the command line features documented in this section are:

- ▶ [Command Line Editing](#)
- ▶ [Command History and Recall](#)
- ▶ [Command History Window](#)
- ▶ [Local and Global History Lists](#)

- [Command Names and Parameters](#)
- [Conditional Expressions](#)
- [Filename Completion](#)
- [Customizing Filename Completion](#)
- [Directory History Window](#)
- [Filename Completion Window](#)
- [Variable Completion](#)
- [Automatic Directory Changes](#)
- [Directory History Window](#)
- [Multiple Commands](#)
- [Expanding and Disabling Aliases](#)
- [Command Line Length Limits](#)
- [Command Grouping](#)
- [Starting Applications](#)
- [Command Parsing](#)
- [Date Formats](#)

Additional command line features are documented under [File Selection](#) and under [Directory Navigation](#).

#### 4.5.1 Command Line Editing

The command line works like a single-line word processor, allowing you to edit any part of the command at any time before you press [Enter](#) to execute it (or [Esc](#) to erase it).

The command line as typed can contain up to a maximum of 65,535 characters, and it can expand to a maximum of 131,071 characters after variable, function and alias substitution. See [Command Line Length Limits](#).

You can use the following editing keys (among others) when you are typing a command (the words **Ctrl** and **Shift** mean to press the *Ctrl* or *Shift* key together with the other key named). The keystrokes listed here are the default values, but most editing keys can be redefined via [Command Line Editing Keys](#) or [General Input Keys](#) directives.

##### Cursor Movement Keys:

|                            |                                                 |
|----------------------------|-------------------------------------------------|
| <a href="#">Left</a>       | Move the cursor left one character              |
| <a href="#">Right</a>      | Move the cursor right one character             |
| <a href="#">Ctrl-Left</a>  | Move the cursor left one word                   |
| <a href="#">Ctrl-Right</a> | Move the cursor right one word                  |
| <a href="#">Home</a>       | Move the cursor to the beginning of the command |
| <a href="#">End</a>        | Move the cursor to the end of the command       |

##### Insert and Delete Keys:

|                                     |                                                                                     |
|-------------------------------------|-------------------------------------------------------------------------------------|
| <a href="#">Ins</a>                 | Toggle between insert and overstrike mode (cursor shape indicates mode)             |
| <a href="#">Del</a>                 | Delete the character under (or to the right of) the cursor, or the highlighted text |
| <a href="#">Bksp</a>                | Delete the character to the left of the cursor, or the highlighted text             |
| <a href="#">Ctrl-L</a>              | Delete the word or partial word to the left of the cursor                           |
| <a href="#">Ctrl-R or Ctrl-Bksp</a> | Delete the word or partial word to the right of the cursor                          |
| <a href="#">Ctrl-Home</a>           | Delete from the beginning of the line to the cursor                                 |
| <a href="#">Ctrl-End</a>            | Delete from the cursor to the end of the line                                       |
| <a href="#">Esc</a>                 | Delete the entire line                                                              |
| <a href="#">Ctrl-V</a>              | Paste the first line of text from the clipboard at the current cursor position      |
| <a href="#">Ctrl-B</a>              | Paste the last argument from the previous command line                              |

|                      |                                                                 |
|----------------------|-----------------------------------------------------------------|
| Ctrl-0 to Ctrl-9     | Paste the corresponding argument from the previous command line |
| <a href="#">Redo</a> | Redo the previous undo                                          |
| <a href="#">Undo</a> | Undo the last edit                                              |

**Highlighting:**

|                        |                                                            |
|------------------------|------------------------------------------------------------|
| Shift-Right            | Highlight character right of cursor and move cursor        |
| Shift-Left             | Highlight character left of cursor and move cursor         |
| Shift-Home             | Highlight from cursor to beginning-of-line and move cursor |
| Shift-End              | Highlight from cursor to end-of-line and move cursor       |
| Ctrl-Shift-Right       | Highlight word right of cursor and move cursor             |
| Ctrl-Shift-Left        | Highlight word left of cursor and move cursor              |
| <a href="#">Ctrl-Y</a> | Copy highlighted text to the clipboard                     |

**Execution:**

|                                                                               |                                                                                                         |
|-------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| <a href="#">Ctrl-K</a>                                                        | Save the current command line in the history list without executing it, and then clear the command line |
| <a href="#">Ctrl-C</a> or <a href="#">Ctrl-Break</a><br><a href="#">Enter</a> | Cancel the command line without saving in the history list<br>Execute the command line                  |

**Miscellaneous:**

|                                                                                     |                                                                                                                             |
|-------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <a href="#">F1</a>                                                                  | Get help for the command (first argument on the line)                                                                       |
| <a href="#">Ctrl-F1</a>                                                             | Get help for the current word                                                                                               |
| Alt-F1                                                                              | Call the command dialog for the command (only argument on the line)                                                         |
| <a href="#">Ctrl-F</a>                                                              | Expand a command or directory alias                                                                                         |
| <a href="#">Ctrl-X</a>                                                              | Expand an environment variable                                                                                              |
| <a href="#">Ctrl-A</a>                                                              | Toggle between LFN and SFN                                                                                                  |
| Alt-PgUp, Alt-PgDn,<br>Alt-Home, Alt-End, Alt-numeric keypad keys.)<br>Up, Alt-Down | Scroll the window within the console buffer. (Use the cursor pad keys, not the Alt-Home, Alt-End, Alt-numeric keypad keys.) |

To highlight text on the command line use the mouse or hold down the **Shift** key and use any of the cursor movement keys listed above. You can select a complete word by placing the cursor anywhere in the word and double-clicking with the mouse. Once you have selected or highlighted text on the command line, any new text you type will replace the highlighted text. If you press **Bksp** or **Del** while there is text highlighted on the command line, the highlighted text will be deleted.

While you are working at the prompt you can use the clipboard to copy text between **TCC** and other applications (see [Highlighting and Copying Text](#) for additional details). You can also use [Drag and Drop](#) to paste a filename from another application onto the command line.

Most of the command line editing capabilities are also available when you are prompted for a line of input. For example, you can use the command line editing keys when **DESCRIBE** prompts for a file description, when **INPUT** prompts for input from an alias or batch file, or when **LIST** prompts you for a search string.

If you want your input at the command line to be in a different color, you can use the [Windows tab](#) of the [configuration dialogs](#).

**TCC** will prompt for additional command line text when you include the escape character as the very last character of a typed command line. (The default escape character is the caret "^", but you can also use

the % = variable). For example:

```
echo The quick brown fox jumped over the lazy ^
More? sleeping dog. > alphabet
```

Sometimes you may want to enter one of the command line editing keystrokes on the command line instead of performing the key's usual action. For example, suppose you have a program that requires a Ctrl-R character on its command line. Normally you couldn't type this keystroke at the prompt, because it would be interpreted as a "Delete word right" command. To get around this problem, use the special keystroke [Alt-255](#). You enter **Alt-255** by holding down the **Alt** key while you type 0255 on the numeric keypad, then releasing the **Alt** key. This forces **TCC** to interpret the next keystroke literally and place it on the command line, ignoring any special meaning it would normally have as a command line editing or history keystroke. You can use **Alt-255** to suppress the normal meaning of command line editing keystrokes even if they have been reassigned with [key mapping directives](#), and **Alt-255** itself can be reassigned with the [CommandEscape](#) configuration option.

#### Alternative Keyboard Input Method:

The method mentioned above for **Alt-255** can be used to generate other characters. You must use the number keys on the numeric keypad, not the row of keys at the top of your keyboard. When this **Alt + keypad** approach is used in a Unicode environment, **TCC** will assume that a 3-digit decimal value means an ASCII character, while a 4-digit decimal value mean a Unicode glyph. Make sure that your hardware, character set, code page and font all support the desired combination. Use caution with this method if you plan on manipulating the generated character in other Windows components. See the section on [ASCII, Key Codes and ANSI X3.64 Commands](#) for some additional information.

### 4.5.2 Command History and Recall

Each time you execute a command, the entire command line is saved in a command history list. You can display the saved commands, search the list, modify commands, and rerun commands. The command history is available at the command prompt and in a special [command history window](#). You can choose to use either a [local or global command history](#).

#### Command History Keys:

|                   |                                                                                                                                                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Up</b>         | Recall the previous (or most recent) command, or the most recent command that matches a partial command line.                                                             |
| <b>Down</b>       | Recall the next (or oldest) command, or the oldest command that matches a partial command line.                                                                           |
| <b>PgUp</b>       | Display a popup window of the command history (or all entries matching a partial command line).                                                                           |
| <b>F3</b>         | Fill in the rest of the command line from the previous command, beginning at the current cursor position.                                                                 |
| <b>Ctrl-D</b>     | Delete the currently displayed history list entry, erase the command line, and display the previous (matching) history list entry.                                        |
| <b>Ctrl-E</b>     | Display the last entry in the history list.                                                                                                                               |
| <b>Ctrl-K</b>     | Save the current command line in the history list without executing it, and then clear the command line.                                                                  |
| <b>Ctrl-Enter</b> | Copy the current command line to the end of the history list even if it has not been altered, then execute it.                                                            |
| <b>@</b>          | As the first character in a line: Do not save the current line in the history list when it is executed, nor store it in the <a href="#">CMDLINE</a> environment variable. |

**Note:** The keystrokes shown above are the default values. See [Key Mapping Directives](#) for details on how to assign different keystrokes.

The simplest use of the command history list is to repeat a command exactly. For example, you might enter the command

```
dir a:*.wks;*.doc
```

to see some of the files on drive A. You might move some new files to drive A and then want to repeat the [DIR](#) command. Just press **Up** repeatedly to scan back through the history list. When the [DIR](#) command appears, press **Enter** to execute it again. You can also view the [command history in a window](#).

After you have found a command, you can edit it before pressing **Enter**. You will appreciate this feature when you have to execute a series of commands that differ only slightly from each other. You can also view and manage the command history list with the [HISTORY](#) command.

The history list is normally "circular". If you move to the latest command in the list and then press **Up** once more, you'll see the oldest command in the list. Similarly, if you move to the first command in the list and then press **Up** once more, you'll see the last command in the list. You can disable this feature and make command history recall stop at the beginning or end of the list by turning off History Wrap on the "History" tab of the configuration dialog.

You can search the command history list to find a previous command quickly using command completion. Just enter the first few characters of the command you want to find and press **Up**. You only need to enter enough characters to identify the command that you want to find. For example, to find a DIR command, enter DI and then press **Up**. If you press **Up** a second time, you will see the previous command that matches. The system will beep if there are no matching commands. The search process stops as soon as you type one of the editing keys, whether or not the line is changed. At that point, the line you're viewing becomes the new line to match if you press **Up** again.

The command history search supports [wildcards](#). For example, you can search for a previous command that contained the string **foo** by typing **\*foo\*** on the command line and pressing the up or down keys.

You can specify the size of the command history list on the **Command Line** tab of the configuration dialog. When the list is full, the oldest commands are discarded to make room for new ones. You can also use the **Minimum Length** option to enable or disable history saves and to specify the shortest command line that will be saved.

You can prevent any command line from being saved in the history by beginning it with an at sign (@) or by including it in the contents of the [HistoryExclude](#) variable.

When you execute a command from the history, that command remains in the history list in its original position. The command is not copied to the end of the list (unless you modify it). If you want each command to be copied or moved to the end of the list when it is reexecuted, select Copy to End or Move to End on the "History" tab of the configuration dialogs. If you select either of these options, the list entry identified as "current" (the entry from which commands are retrieved when you press **Ctrl-Up**) is also adjusted to refer to the end of the history list after each recalled command is executed.

Use **F3** when your new command is different from your previous one by just a character or two at the beginning. For example, suppose you want to execute a **DIR** on several file names then use **DEL** to delete those same files. After the **DIR** is complete type **DEL** and press **F3**; the rest of the command line

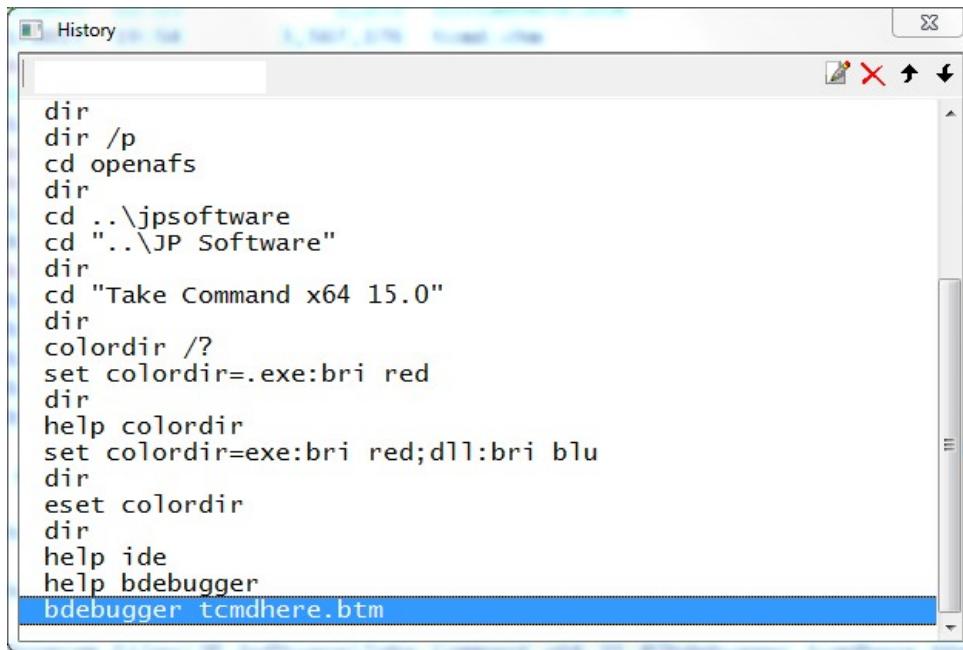
will be completed for you. Check that it's correct, and then press **Enter** to delete the files. **F3** also retrieves the entire previous command (like **Up**) if nothing has been typed on the line.

Use **Ctrl-E** to "get your bearings" by returning to the end of the list if you've scrolled around so much that you aren't sure where you are any more.

Use **Ctrl-K** to save some work when you've typed a long command and then realize that you weren't quite ready. For example, if you forgot to change directories and notice it after a command is typed or mostly typed, but before you press **Enter**, just press **Ctrl-K** to save the command without executing it. Use the **CD** or **CDD** command to change to the right directory, press **Up** twice to retrieve the command you saved, make any final changes to it, and press **Enter** to execute it.

Use **Ctrl-Enter** to organize the history list for repetitive tasks. Instead of searching through the command history for the next command in a sequence, you can place all of the necessary commands next to each other and make them easier to repeat.

#### 4.5.3 Command History Window



You can view the command history in a scrollable popup window, and select the command to re-execute or modify from those displayed in the window. The directory history window includes a toolbar with buttons for editing, deleting, and moving lines.

To activate the command history window press **PgUp** or **PgDn** at the command line. A popup window will appear, with the command you most recently executed marked with a highlight. (If you just finished re-executing a command from the history, then the next command in sequence will be highlighted.)

You can view a "filtered" history window by typing some characters on the command line, then pressing **PgUp** or **PgDn**. Only those commands matching the typed characters will be displayed in the window.

You can search for a specific command by entering a string (including wildcards or regular expressions) in the edit window on the title bar. **TCC** will remove non-matching lines from the window. See [Popup](#)

[Windows](#) for details.

#### Command History Window Keys:

|                               |                                                        |
|-------------------------------|--------------------------------------------------------|
| Up                            | Scroll the display up one line.                        |
| Down                          | Scroll the display down one line.                      |
| <a href="#">Left</a>          | Scroll the display left 4 columns.                     |
| <a href="#">Right</a>         | Scroll the display right 4 columns.                    |
| PgUp                          | Scroll the display up one page.                        |
| PgDn                          | Scroll the display down one page.                      |
| Home                          | Go to the beginning of the list.                       |
| End                           | Go to the end of the list.                             |
| <a href="#">Ctrl-Enter</a>    | Move the selected line to the command line for editing |
| <a href="#">Enter</a>         | Execute the selected line                              |
| Ctrl-C                        | Copy the selected line to the clipboard                |
| <a href="#">Ctrl-D</a> or Del | Delete the selected line from the list                 |
| Ctrl-E                        | Edit the selected line in the history window           |
| Ctrl-Up                       | Move the selected line up one row                      |
| Ctrl-Down                     | Move the selected line down one row                    |
| <a href="#">Esc</a>           | Close the window without making a selection.           |

**Note:** The keystrokes shown above are the default values. See [Key Mapping Directives](#) for details on how to assign different keystrokes.

Once you have selected a command in the history window, press **Enter** or double-click with the mouse to execute it immediately. Press **Ctrl-Enter** or hold down the Ctrl key while you double-click with the mouse to move the line to the prompt for editing.

You can move and/or resize the history window. **TCC** will save the changes and use the new position the next time the command history window is invoked. (Not supported in **TCC/LE**.) You can also change the keys used in the window with [key mapping directives](#).

#### 4.5.4 Local and Global History Lists

The [command history](#) and [directory history](#) can be stored in either local or global lists.

With a local list, any changes made to the history will only affect the current **TCC** tab window. They will not be visible in other tabs or other copies of **TCC**.

With a global list, all **TCC** windows will share the same history, and any changes made to the history in one copy (e.g., by executing commands from the prompt) will affect all other copies. Global lists are the default.

You can control the type of history list with the [Local History](#) and [Local Directory History](#) options, and with the **/L**, **/LD** and **/LH** options of either the [START](#) command or **TCC**.

If you select a global history list for **TCC**, you can share the history among all **TCC** sessions running concurrently. When you close all of the **TCC** sessions, the memory for the global history list is released, and a new, empty history list is created the next time you start **TCC**.

If you want the histories to be retained in memory even when no **TCC** session is running, see the [SHRALIAS](#) command, which retains the global alias, user-defined function, command history, and directory history lists. [SHRALIAS](#) retains the lists in memory, but cannot preserve it when Windows itself is shut down or the user logs out. To save your histories for the next restart of Windows, you must store

them in a file and reload them after the system restarts. For details on how to do so, see the [HISTORY](#) and [DIRHISTORY](#) commands.

#### 4.5.5 Command Names & Parameters

When you enter a command you type its name at the prompt, followed by a space and any parameters for the command. For example, all of these could be valid commands:

```
dir
copy file1 file2 d:\
f:\util\mapmem /v
"c:\program files\JPSoft\tcmd13\tcc.exe" /LF
```

The last three commands above include both a command name, and one or more parameters. There are no spaces within the command name (except in quoted file names), but there is a space between the command name and any options or parameters, and there are spaces between the options and parameters.

Some commands may work when options or parameters are entered directly after the command (without an intervening space, e.g. dir/p), or when several options or parameters are entered without spaces between them (e.g. dir /2/p). A very few older programs may even require this approach. However, leaving out spaces this way is usually technically incorrect, and is not recommended as a general practice, as it may not work for all commands.

If the command name includes a path, the elements must be separated with backslashes (e.g. F:\UTIL\MAPMEM). If you are accustomed to Linux syntax where forward slashes are used in command paths, and want **TCC** to recognize this approach, you can set the [Unix/Linux-style Paths](#) option.

For more information on command entry see [Multiple Commands](#) and [Command Line Length Limits](#). For details on how **TCC** handles the various elements it finds on the command line see [Command Parsing](#).

#### 4.5.6 Conditional Expressions

The commands [DO](#) (when used with the UNTIL or WHILE keyword), [IF](#), [IFF/ELSEIFF](#), and the variable function [@IF](#) evaluate a conditional expression, and perform a different action based on whether or not the expression is TRUE. The [SWITCH](#) command tests pairs of values for equality. Most of the [examples](#) below use the [IF](#) command, but conditional expressions could be used in the other cases above as well.

A conditional expression can be one of the following, as described below:

- [relational expression](#)
- [status test](#)
- [logical expression](#)

##### Relational Expression

A relational expression compares two character strings, using one of the [relational operators](#) in the table below. Each of these two character strings can contain literal text, environment and internal variables, and variable functions, including user defined ones, in any combination. Note that double quotes are significant.

##### Numeric and String Comparison

When comparing the two character strings, either a numeric or a string comparison will be used. A

numeric comparison treats the strings as numeric values and tests them arithmetically. A string comparison treats the strings as text. The parser uses the rules described for the [@NUMERIC](#) function to determine whether or not the strings are numeric, and only if both are numeric is a numeric comparison performed. If either value is non-numeric, a string comparison is used. To force a string comparison when both values may be numeric, use double quotes around the values you are testing, as shown below. Because the quote mark is not a numeric character, string comparison is performed. Numeric comparison cannot be forced. To compare hexadecimal numbers numerically, you must convert them to decimal numbers using [@CONVERT](#). This is not necessary if both are the same length - string comparison and numeric comparison yield the same result.

The example below demonstrates the difference between numeric and string comparisons, as shown in the table below. Numerically, 2 is smaller, but as a string it is "larger" because its first digit is larger than the first digit of 19. So the first of these conditions will be true, and the second will be false:

| expression  | value | comparison type |
|-------------|-------|-----------------|
| 2 lt 19     | true  | numeric         |
| "2" lt "19" | false | string          |

### Relational Expression Formats

The format of a relational expression is one of

*num1* [relational operator](#) *num2*  
*string1* [relational operator](#) *string2*

**Note:** The correct syntax requires a space both before and after **operator** to separate it from its operands. Commonly seen constructs such as %a==b may or may not work depending on the specific parameters, but they are *never* recommended.

### Relational Operators

|                        |                                                         |                                                                                       |
|------------------------|---------------------------------------------------------|---------------------------------------------------------------------------------------|
| <b>operator</b>        | numeric comparison: <i>expression</i> is <b>true</b> if | string comparison: <i>expression</i> is <b>true</b> if, when ignoring character case: |
| <b>EQ</b> or <b>==</b> | <i>num1</i> equals <i>num2</i>                          | <i>string1</i> equals <i>string2</i>                                                  |
| <b>NE</b> or <b>!=</b> | <i>num1</i> does not equal <i>num2</i>                  | <i>string1</i> does not equal <i>string2</i>                                          |
| <b>LT</b>              | <i>num1</i> is less than <i>num2</i>                    | <i>string1</i> alphabetically precedes <i>string2</i>                                 |
| <b>LE</b>              | <i>num1</i> is less than or is equal to <i>num2</i>     | <i>string1</i> alphabetically precedes or is equal to <i>string2</i>                  |
| <b>GE</b>              | <i>num1</i> is greater than or is equal to <i>num2</i>  | <i>string1</i> alphabetically succeeds or is equal to <i>string2</i>                  |
| <b>GT</b>              | <i>num1</i> is greater than <i>num2</i>                 | <i>string1</i> alphabetically succeeds <i>string2</i>                                 |
| <b>EQC</b>             | tested as strings →                                     | <i>string1</i> is identical to <i>string2</i> , including character case              |
| <b>=~</b>              | regular expression test                                 | <i>string1</i> matches the regular expression in <i>string2</i>                       |
| <b>!~</b>              | regular expression test                                 | <i>string1</i> doesn't match the regular expression in <i>string2</i>                 |

Case differences are ignored in string comparisons (except by **EQC**). If two strings begin with the same text but one is shorter, the shorter string is considered to precede (be less than) the longer one. For example, "a" is less than "abc", and "hello\_there" is greater than "hello".

When you compare text strings, you may need to enclose the parameters in double quotes in order to avoid syntax errors which can occur if one of the parameter values is empty (e.g., due to an environment variable which has never been assigned a value). This technique will not work for numeric comparisons, as the quotes will force a string comparison, so with numeric tests you must be sure that all variables are assigned values before the test is done.

In order to maintain compatibility with CMD, **TCC** recognizes the following additional names for conditions:

| CMD        | TCC |
|------------|-----|
| EQL or EQU | EQ  |
| NEQ        | NE  |
| LSS        | LT  |
| LEQ        | LE  |
| GTR        | GT  |
| GEQ        | GE  |

[Internal variables](#) and [variable functions](#) are very powerful when combined with string and numeric comparisons. They allow you to test the state of your system, the characteristics of a file, date and time information, or the result of a calculation. You may want to review the variables and variable functions when determining the best way to set up a condition test.

### Status Test

These conditions test operating system, file system or **TCC** status. In addition to the tests below, there are many internal variables and variable functions which allow you to test the status of many other parts of the system.

In the descriptions below of the various status tests, the status tests are true if and only if the specified condition is true.

#### **DEFINED** *variable*

If variable exists in the environment, the expression is true. This is equivalent to testing whether or not variable is nonempty.

Note: [GOSUB variables](#), [array variables](#), and [internal variables](#) do not exist in the environment, so they always fail the DEFINED test.

#### **ERRORLEVEL** [[relational operator](#)] n

This test retrieves the exit code of the preceding external program. By convention, programs return an exit code of 0 when they are successful and a non-zero number to indicate an error. The [relational operator](#) may be any of those listed above (e.g., EQ, GT). If no operator is specified, the default is GE. The comparison is done numerically.

Not all programs return an explicit exit code. For programs which do not, the behavior of ERRORLEVEL is undefined.

#### **EXIST** *filename*

If filename matches a file which exists, the expression is true. You can use wildcards in filename, in which case the expression is true if any file matching the wildcard name exists. filename may include an absolute or relative path.

**WARNING:** In Windows the expression will be true if there is either a

file or a directory named filename. Use ISFILE or ISDIR instead.

The special filename NUL is commonly used in CMD batch files to test the existence of a directory. The expression exist xxx\NUL is true only if xxx is a directory.

**ISALIAS *aliasname***

If aliasname is defined as an alias, the expression is true.

**ISAPP *appname***

If appname matches the name of an application which is currently running, the expression is true. To match a specific application, you must enter the full pathname of the application. Partial names and wildcards will yield undependable results. Both the short and long filename forms of the name will be checked (see [LFN File Searches](#) for details on the correspondence between short and long filenames). This test may require DEBUG privilege.

**ISDIR *path***

**DIREXIST *path***

If the directory specified by path exists, the expression is true. Path may be either absolute or relative. DIREXIST may be used as a synonym for ISDIR.

**ISFILE *filename***

If filename matches a file which exists, the expression is true. You can use wildcards in the filename, in which case the expression is true if any file matching the wildcard name exists. ISFILE matches only files, not directories.

**ISFUNCTION *name***

If the user-defined function name is loaded, the expression is true.

**ISINTERNAL *command***

If command is an active internal command, the expression is true. Commands can be activated and deactivated with the [SETDOS /I](#) command.

**ISLABEL *label***

If label exists in the current batch file, the expression is true. Labels may be one or more words long. Note that this test has nothing to do with disk partition labels.

**ISPLUGIN *name***

If name is a [plugin](#) variable, function, or command, the expression is true.

**ISWINDOW "title"**

If a window which matches the title exists, the expression is true. double quotes must be used around the title, which may contain wildcards and extended wildcards.

**ISHUNG "title"**

If the specified window is not responding, the expression is true.

**PLUGIN *module***

If the plugin module is loaded, the expression is true. Do not include an extension (i.e., ".dll"), for the module name.

## Logical Expressions

A logical expression is one of the following:

- a [relational expression](#)
- a [status test](#)
- the unary logical operator **NOT** (or **!**) followed by a [logical expression](#)

- two [logical expressions](#) connected by a binary logical operator

### Logical operators

| operator     | type   | usage                    | value is TRUE if                                            |
|--------------|--------|--------------------------|-------------------------------------------------------------|
| <b>NOT</b>   | unary  | <b>NOT</b> cond          | cond is FALSE.                                              |
| <b>.AND.</b> | binary | cond1 <b>.AND.</b> cond2 | both cond1 and cond2 are TRUE.                              |
| <b>.OR.</b>  | binary | cond1 <b>.OR.</b> cond2  | at least one of cond1 and cond2 is TRUE.                    |
| <b>.XOR.</b> | binary | cond1 <b>.XOR.</b> cond2 | one of cond1 and cond2 is TRUE, and the other one is FALSE. |

This example runs a program called **DATALOAD** if today is Monday or Tuesday (enter this on one line):

```
if "%_dow" == "Mon" .or. "%_dow" == "Tue" dataload
```

Test conditions are always scanned from left to right -- there is no implied order of precedence, as there is in some programming languages. You can, however, force a specific order of testing by grouping conditions with parentheses, for example (enter this on one line):

```
if (%a == 1 .or. (%b == 2 .and. %c == 3)) echo something
```

### Combining logical expressions

**Parentheses** can be used only when the portion of the **expression** inside the parentheses contains at least one of the binary logical operators **.and.**, **.or.**, or **.xor.**. Parentheses on a simple expression which does not combine two or more tests will be taken as part of the string to be tested, and will probably make the test fail. For example, the first of these tests is **FALSE**, the second is **TRUE**:

```
(a == a)
(a == a .and. b == b)
```

Parentheses may be nested.

### Examples

This batch file fragment runs a program called **WEEKLY** if today is Monday:

```
if "%_dow" == "mon" weekly
```

This batch file fragment tests for a string value:

```
input "Enter your selection : " %%cmd
if "%cmd" == "WP" goto wordproc
if "%cmd" NE "GRAPHICS" goto badentry
```

This example calls **GO.BTM** if the first two characters in the file **MYFILE** are **GO**:

```
if "%@left[2,%@line[myfile,0]]" == "GO" call go.btm
```

The first batch file fragment below tests for the existence of **A:\JAN.DOC** before copying it to drive c (this avoids an error message if the file does not exist):

```
if isfile a:\jan.doc copy a:\jan.doc c:\
```

This example tests the exit code of the previous program and stops all batch file processing if an error occurred:

```
if errorlevel == 0 goto success
echo "External Error; Batch File Ends!"
cancel
```

#### 4.5.7 Filename Completion

Filename completion can help you by filling in a complete file name on the command line when you only remember or want to type part of it. Filename completion can be used at the command line, which is explained here, and in a [filename completion window](#).

##### Filename Completion Keys:

|                        |                                                                                                                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>F8 or Shift-Tab</b> | Get the previous matching filename.                                                                                                                                                                                 |
| <b>F9 or Tab</b>       | Get the next matching filename.                                                                                                                                                                                     |
| <b>F10</b>             | Keep the current matching filename and display the next matching name immediately after the current one.                                                                                                            |
| <b>F12</b>             | Repeat the filename just returned from an F9 / Tab match.                                                                                                                                                           |
| <b>Alt-F9</b>          | Restore the original filename mask after a previous F9 or Tab. (This will only work provided you haven't terminated the completion loop; i.e. by pressing anything other than Tab, Shift-Tab, F8, F9, F10, or F12.) |
| <b>Ctrl-A</b>          | Toggle between long and short filename.                                                                                                                                                                             |

**Note:** The keystrokes shown above are the default values. See [Key Mapping Directives](#) for details on how to assign different keystrokes.

For example, if you know the name of a file begins *AU* but you can't remember the rest of the name, type:

```
copy au
```

and then press the **Tab** key or **F9** key. **TCC** will search the current directory for filenames that begin with *AU* and insert the first one onto the command line in place of the *AU* that you typed.

If this is the file that you want, simply complete the command. If **TCC** didn't find the file that you were looking for, press **Tab** or **F9** again to substitute the next filename that matches your pattern (in the above example, begins with *AU*). When there are no more filenames that match your pattern, the system will beep each time you press **Tab** or **F9**.

If you go past the filename that you want, press **Shift-Tab** or **F8** to back up and return to the previous matching filename. After you back up to the first filename, the system will beep each time you press **Shift-Tab** or **F8**.

If you want to enter more than one matching filename on the same command line, press **F10** when each desired name appears. This will keep that name and place the next matching filename after it on the command line. You can then use **Tab** (or **F9**) and **Shift-Tab** (or **F8**) to move through the remaining matching files.

The pattern you use for matching may contain any valid filename characters, as well as wildcard characters and extended [wildcards](#). For example, you can copy the first matching *.TXT* file by typing

```
copy *.txt
```

and then pressing **Tab**.

If you don't specify part of a filename before pressing **Tab**, **TCC** will match all files. For example, if you enter the above command as "COPY ", without the *.TXT*, and then press **Tab**, the first filename in the current directory is displayed. Each time you press **Tab** or **F9** after that, another name from the current directory is displayed, until all filenames have been displayed. **Note:** you must terminate the command (e.g., by space) before file completion becomes available.

**TCC** will append \* to the name on LFN drives, and \*.\* on drives which only support short file names. If you are typing a group of file names in an [include list](#), the part of the include list at the cursor will be used as the pattern to match.

When filename completion is used at the start of the command line, it will only match directories, executable files, and files with [executable extensions](#), since these are the only file names that it makes sense to use at the start of a command. If a directory is found, a \ will be appended to it to enable an [automatic directory change](#). If you need to complete the name of any other file at the start of the command line, press **Space** before starting to type the name. Filename completion will then match any name, not just directory and executable names. Note that you can also "execute" files whose extension has an association in the Windows Registry, but such files are not considered executable by **TCC**, and only the method above using a space will work.

Filename completion occurs in the physical order in which matching filenames are stored in the directory, the same order in which [DIR /O:U](#) would list them. That order is determined by the underlying file system.

**TCC** also supports network server and sharename completion. If the filename begins with \\, the completion routines will enumerate the network resources for matching server and/or share names. You can control the way server name completion functions with the [Server Completion](#) configuration option.

Filename completion will search the [PATH](#) for an executable filename if you have set the **Search Path** option in the **Command Line** configuration tab, and you are :

- (1) at the beginning of the command line, and
- (2) there are no matching entries in the current directory, and
- (3) the name you are attempting to match doesn't contain a full or partial path specification.

If all three conditions are met, filename completion will return the first matching executable found in the [PATH](#).

If you are on an NTFS drive, you can also complete stream names. For example:

```
copy test:t
```

and then pressing **Tab** will search the file **test** for streams beginning with "t". Note that you cannot complete a filename and a stream name simultaneously (i.e., **t\*:t\***).

Several topics are related to filename completion. See:

- › [Converting Between Long and Short Filenames](#)
- › [Appending Backslashes to Directory Names](#)
- › [Customizing Filename Completion](#)
- › [Filename Completion Window](#)
- › [Variable Name Completion](#)

#### 4.5.8 Customizing Filename Completion

You can customize filename completion for any internal or external command or alias. This allows **TCC** to display filenames intelligently based on the command you are entering. For example, you might want to see only **.TXT** files when you use filename completion in the **EDIT** command.

To customize filename completion you can use the [Filename Completion](#) configuration options. You can also use the [FILECOMPLETION](#) environment variable. If you use both, the environment variable will override the configuration option. You may find it useful to use the environment variable for experimenting, then create permanent settings with the configuration dialog.

The format for both the environment variable and the directive is:

```
cmd1 [cmd2 ...]:[!]ext1 ext2 ...; cmd2: ...
```

where

**cmd1** etc. are command names

**ext1** etc. are file extensions (which may include wildcards) or one of the following file types:

|                |                                      |
|----------------|--------------------------------------|
| <b>DIRS</b>    | Directories                          |
| <b>RDONLY</b>  | Read-only files                      |
| <b>HIDDEN</b>  | Hidden files                         |
| <b>SYSTEM</b>  | System files                         |
| <b>ARCHIVE</b> | Files modified since the last backup |
| <b>FILES</b>   | Everything that's not a directory    |

Note that if a file uses one of the reserved file type names shown above as its extension (e.g. *xyz.hidden*), that file will be treated as if it were of that type.

Setting options in OPTION / Filename Completion, or with the FILECOMPLETION environment variable, will override the default filename completion settings such as "complete hidden files / directories" options for that command. If you want to customize filename completion **and** search for hidden / system files, you will need to add the HIDDEN and/or SYSTEM extensions to that command's filename completion options.

You can exclude an extension by prefixing it with a **!**.

The command name is the internal command, alias, or executable file name (without a path). For example, to have file completion return only directories for the [CD](#), [CDD](#), and [RD](#) commands and only **.C** and **.ASM** files for a Windows editor called WinEdit, you would use this setting for filename completion in the configuration dialog:

```
cd cdd rd:dirs; winedit:c asm
```

To set the same results using the [FILECOMPLETION](#) environment variable:

```
set filecompletion=cd cdd rd:dirs; winedit:c asm
```

With this setting in effect, if you type "CD " and then pressed **Tab**, **TCC** returns only directories, not files. If you type **WINEDIT** and press **Tab**, you will see only names of .C and .ASM files.

When testing for a customized filename match, **TCC** checks the actual command line you type (but without expanding any aliases). For example, if you use the definition above and have "W" aliased to "WINEDIT" and then enter a "W" command, filename completion -- which refers only to "WINEDIT" -- will be ignored. To use customized filename completion for aliases you must enter the alias name:

```
FileCompletion=cd cdd rd:dirs; winedit:c asm; w:c asm
```

#### 4.5.9 Filename Completion Window

You can view matching filenames in a filename completion window. To activate the window, press **F7** or **Ctrl-Tab** at the command line. You will see a popup window, with a sorted list of files that match any partial filename you have entered on the command line. If you haven't yet entered a file name, the window will contain the name of all executable files (or files with an association; see [ASSOC](#)) in the current directory. You can search for a name by entering a string (including wildcards or regular expressions) in the edit window on the title bar. **TCC** will remove non-matching lines from the window. See [Popup Windows](#) for details.

##### Filename Completion Window Keys:

|                                       |                                                     |
|---------------------------------------|-----------------------------------------------------|
| F7 or <a href="#">Ctrl-Tab</a>        | (from the command line) Open the window.            |
| Up                                    | Scroll the display up one line.                     |
| Down                                  | Scroll the display down one line.                   |
| <a href="#">Left</a>                  | Scroll the display left 4 columns.                  |
| <a href="#">Right</a>                 | Scroll the display right 4 columns.                 |
| PgUp                                  | Scroll the display up one page.                     |
| PgDn                                  | Scroll the display down one page.                   |
| Home                                  | Go to the beginning of the list.                    |
| End                                   | Go to the end of the list.                          |
| <a href="#">Enter</a> or Double Click | Insert the selected filename into the command line. |

**Note:** The keystrokes shown above are the default values. See [Key Mapping Directives](#) for details on how to assign different keystrokes.

See also: [Filename Completion](#)

#### 4.5.10 Converting Between Long & Short Filenames

On LFN drives, **TCC** will search for and display long filenames during filename completion. If you want to search for 8.3 short filenames (SFNs), press [Ctrl-A](#) before you start using filename completion. This allows you to use filename completion on LFN drives with applications that do not support long filenames. The [LFNToggle](#) directive can be used to change the keystroke assigned to this feature.

You can press **Ctrl-A** at any time prior to beginning filename completion. The switch to SFN format remains in effect for the remainder of the current command line. When **TCC** begins a new command line it returns to long filename format until you press **Ctrl-A** again.

You can also press **Ctrl-A** just after a filename is displayed, and the name will be converted to short filename format. However, this feature only affects the most recently entered file or directory name (the part between the cursor and the last backslash [\] on the command line), and any subsequent entries. It will not automatically convert all the parts of a previously entered path.

**Ctrl-A** toggles the filename completion mode, so you can switch back and forth between long and short filename displays by pressing **Ctrl-A** each time you want to change modes.

#### 4.5.11 Appending Backslashes to Directory Names

If you set the **Add \ to Directories** option in the Command Line tab of the configuration dialogs, **TCC** will add a trailing backslash \ to directory names. The character appended is a slash / for directory names in [FTP URLs](#) or (if you have set the **UNIX/Linux-style Paths** option in the Startup tab) to all directory names.

This feature can be especially handy if you use filename completion to specify files that are not in the current directory. A succession of **Tab** or **F9** and **F10** keystrokes can build a complete path to the file you want to work with.

The following example shows the use of this technique to edit the file *C:\DATA\FINANCE\MAPS.DAT*. The lines which include "<F9>" show where F9 (or Tab) is pressed; the other lines show how the command line appears after the previous F9 or Tab (the example is displayed on several lines here, but all appears at a single command prompt when you actually perform the steps):

```
1 edit \da <F9>
2 edit \data\
3 edit \data\f <F9>
4 edit \data\frank.doc <F9>
5 edit \data\finance\
6 edit \data\finance\map <F9>
7 edit \data\finance\maps.dat
```

Note that F9 was pressed twice in succession on lines 3 and 4, because the file name displayed on line 3 was not what was needed. We were looking for the *FINANCE* directory, which came up the second time F9 was pressed.

#### 4.5.12 Extended Parent Directory Names

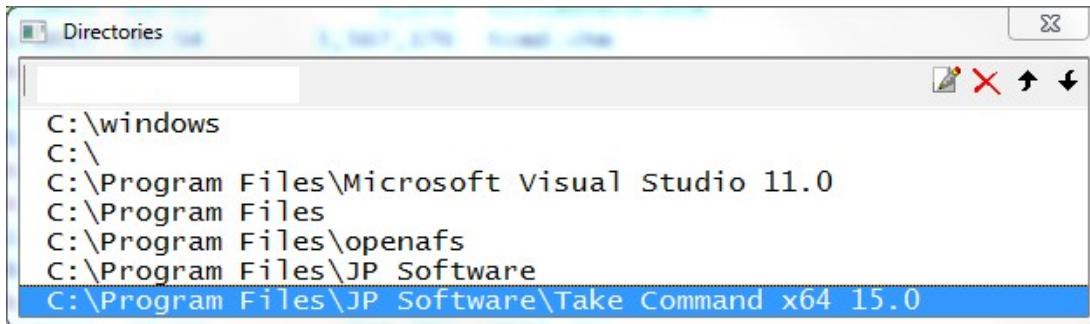
**TCC** has an extended syntax for referencing parent directories, by adding additional . characters. Each additional . represents an additional directory level above the current directory. For example, *.FILE.DAT* refers to a file in the current directory, *..FILE.DAT* refers to a file one level up, i.e., in the parent directory, and *...FILE.DAT* refers to a file two levels up, i.e., in the parent of the parent directory. If your default directory is *C:\DATA\FINANCE\JANUARY*, you can copy the file *LETTERS.DAT* from directory *C:\DATA* to drive *A:* with the command

```
[C:\DATA\FINANCE\JANUARY] copy ...\\LETTERS.DAT A:
```

**Note:** This extended notation may not be understood by external programs. Consider using the [@FULL](#) function to expand file and directory references when necessary:

```
[C:\DATA\FINANCE\JANUARY] myprog %@full[...\\LETTERS.DAT]
```

#### 4.5.13 Directory History Window



[The directory history window is part of a set of comprehensive directory navigation features built into **TCC**. For a summary of these features, and more information on enhanced directory navigation features, see [Directory Navigation](#).]

The directory history window includes a toolbar with buttons for editing, deleting, and moving lines.

You can search for a specific directory by entering a string (including wildcards or regular expressions) in the edit window on the title bar. **TCC** will remove non-matching lines from the window. See [Popup Windows](#) for details.

##### Directory History Window Keys:

|                                 |                                                        |
|---------------------------------|--------------------------------------------------------|
| <a href="#">F6 or Ctrl-PgUp</a> | Open the window from the command line                  |
| Up                              | Scroll the display up one line.                        |
| Down                            | Scroll the display down one line.                      |
| <a href="#">Left</a>            | Scroll the display left 4 columns.                     |
| <a href="#">Right</a>           | Scroll the display right 4 columns.                    |
| PgUp                            | Scroll the display up one page.                        |
| PgDn                            | Scroll the display down one page.                      |
| Home                            | Go to the beginning of the list.                       |
| End                             | Go to the end of the list.                             |
| <a href="#">Ctrl-Enter</a>      | Move the selected line to the command line for editing |
| <a href="#">Enter</a>           | Change to the selected drive/directory                 |
| Ctrl-C                          | Copy the selected line to the clipboard                |
| <a href="#">Ctrl-D or Del</a>   | Delete the selected line from the list                 |
| <a href="#">Ctrl-E</a>          | Edit the selected line in the directory history window |
| Ctrl-Up                         | Move the selected line up one row                      |
| Ctrl-Down                       | Move the selected line down one row                    |
| <a href="#">Esc</a>             | Close the window without making a selection.           |

**Note:** The keystrokes shown above are the default values. See [Key Mapping Directives](#) for details on how to assign different keystrokes.

The current directory is recorded automatically in the directory history list just before each change to a new directory or drive.

You can view the directory history from the scrollable directory history window and change to any drive and directory on the list. To activate the directory history window, press [F6](#) at the command line. You can then select a new directory with the [Enter](#) key or by double-clicking with the mouse.

If the directory history list becomes full, old entries are deleted to make room for new ones. You can set the size of the list with the [Command History Buffer Size](#) configuration option. You can change the keys used in the window with [key mapping directives](#).

In order to conserve space, each directory name is recorded just once in the directory history, even if you move into and out of that directory several times. The directory history can be stored in either a local or global list; see below for details.

When you switch directories, the original directory is saved in the directory history list, regardless of whether you change directories at the command line, from within a batch file, or from within an alias. However, directory changes made by external directory navigation utilities or other external programs are not recorded by **TCC**.

You can also view and manage the directory history list with the [DIRHISTORY](#) command.

### Local and Global Directory History

The directory history can be stored in either a local or global list. With a local directory history list, any changes made to the list will only be known only to the current copy of **TCC**. They will not be visible in other sessions. Whenever you start another shell which uses a local history list, it inherits a copy of the directory history from the previous shell. However, any changes to the history made in the second shell will affect only that shell.

All copies of **TCC** using global directory history list will share a single copy of directory history. Any directory changes made in any of these copies of **TCC** will be recorded in that shared list, and be accessible by all of them. However, any additional copies of **TCC** which use local directory history will see their own local lists. Global lists are the default for **TCC**.

You can control the type of history list with the [Local Directory History](#) configuration option, with the /L and /LD [command line options](#), and with the /L and /LD options of the [START](#) command.

When you close all **TCC** sessions, the memory for the global directory history list is released, and a new, empty directory history list is created the next time you start **TCC**. If you want the directory history list to be retained in memory even when no copy of **TCC** is running, you need to execute the [SHRALIAS](#) command, which performs this service for the global command history, directory history, user-defined functions, and aliases.

There is no fixed rule for deciding whether to use a local or global directory history list. Depending on your work style, you may find it most convenient to use one type, or a mixture of types in different sessions or shells. We recommend that you start with a global directory history, then modify it if you find a situation where the default is not convenient.

#### 4.5.14 Variable Name Completion

Variable name completion works like [filename completion](#). If the parameter begins with a %, the completion routines will scan the environment, internal variables, and variable functions for matching variable names. For example, if the [PROMPT](#) and [PATH](#) variables are in the environment, in that order, and no other variables start with p, the sequence below may be used to display the value of [PATH](#):

```
echo %p<Tab>
echo %PROMPT<Tab>
echo %PATH<Enter>
```

### 4.5.15 Expanding and Disabling Aliases

A few command line options are specifically related to aliases, and are documented briefly here for completeness. If you are not familiar with aliases, see [Aliases](#) and the [ALIAS](#) command for complete details.

You can expand an alias on the command line and view or edit the results by pressing [Ctrl-F](#) before the command is executed. This is useful when you are developing and debugging a complex alias or if you want to make sure that an alias that you may have forgotten won't change the intent of your command.

At times, you may want to temporarily disable an alias that you have defined. To do so, precede the command with an asterisk (\*). For example, if you have an alias for DIR which changes the display format, you can use the following command to bypass the alias and display the directory in the standard format:

```
*dir
```

**Note:** The leading asterisk is crucial in aliases that redefine existing commands, such as:

```
DIR=*dir /w
```

Without the asterisk, you would trigger an **alias loop error** whenever you try to use that alias, since it will endlessly try to redefine itself.

### 4.5.16 Multiple Commands

You will often know the next two or three commands that you want to execute. Instead of waiting for each one to finish before you type the next, you can type them all on the same command line, separated by the command separator (by default, an ampersand &) or the %+ pseudovariable. For example, if you know you want to copy all of your .TXT files to D:\TEXT and then delete all of them beginning with 'A', you could enter the following command:

```
copy *.txt d:\text\ & del a*.txt
```

You may put as many commands on the command line as you wish, as long as the total length of the command line does not exceed 65,535 characters before alias and variable expansion, and 131,071 characters after expansion.

You can use multiple commands in [alias](#) definitions and [batch files](#) as well as from the command line.

If you don't like using the default command separator, you can pick another character using the [SETDOS](#) command's /C option, or the [Separator character](#) configuration option.

### 4.5.17 Conditional Commands

When an internal command or external program finishes, it returns a result called the [exit code](#). Conditional commands allow you to perform tasks based upon the previous command's [exit code](#). Many programs return 0 if they are successful and a non-zero value if they encounter an error.

#### AND operator &&

If you separate two commands by **&&** (AND), the second command will be executed only if the first command's [exit code](#) is 0. For example, the following command will only erase files if the BACKUP

operation succeeds:

```
backup c:\ a: && del c:*.bak;*.lst
```

### OR operator ||

If you separate two commands by || (OR), the second command will be executed only if the first command's [exit code](#) is non-zero. For example, if the following BACKUP operation fails, then [ECHO](#) will display a message:

```
backup c:\ a: || echo Error in the backup!
```

All internal commands return an [exit code](#), but not all external programs do. Conditional commands will behave unpredictably if you use them with external programs which do not return an explicit [exit code](#). To determine whether a particular external program returns a meaningful [exit code](#) use an ECHO %? command immediately after the program is finished. If the program's documentation does not discuss [exit code](#), you may need to experiment with a variety of conditions to see how the [exit code](#) changes.

## 4.5.18 Command Grouping

Command grouping allows you to group a set of commands together logically by enclosing them in parentheses.

There are two primary uses for command grouping. One is to execute multiple commands in a place where normally only a single command is allowed. For example, suppose you wanted to execute two different [REN](#) commands in all subdirectories of your hard disk. You could do it like this:

```
global ren *.wx1 *.wxo
global ren *.tx1 *.txo
```

But with command grouping you can do the same thing in one command:

```
global (ren *.wx1 *.wxo & ren *.tx1 *.txo)
```

The two [REN](#) commands enclosed in the parentheses appear to [GLOBAL](#) as if they were a single command, so both commands are executed for every directory, but the directories are only scanned once, not twice, typically saving time.

This kind of command grouping is most useful with the [EXCEPT](#), [FOR](#), [GLOBAL](#), and [IF](#) commands. When you use this approach in a batch file, you must either place all of the commands in the group on one line, or place the opening parenthesis at the end of a line and place the commands on subsequent lines. Examples 1 and 2 below will work properly, but Example 3 will not:

Example 1 (correct):

```
for %f in (1 2 3) (echo hello %f & echo goodbye %f)
```

Example 2 (correct):

```
for %f in (1 2 3) (
 echo hello %f
 echo goodbye %f
)
```

Example 3 (incorrect):

```
for %f in (1 2 3) (echo hello %f
echo goodbye %f)
```

If the above examples are typed at the command line, **TCC** will issue a **More?** prompt in response to each line until the command group is closed (i.e. the final parenthesis is recognized) as discussed below.

The second common use of command grouping is to redirect input or output for several commands without repeatedly using the [redirection](#) symbols. For example, consider the following batch file fragment which places some header lines (including today's date) and directory displays in an output file using redirection. The first [ECHO](#) command creates the file using `>`, and the other commands append to the file using `>>`:

```
echo Data files %_date > filelist
dir *.dat >> filelist
echo. >> filelist
echo Text files %_date >> filelist
dir *.txt >> filelist
```

Using command grouping, these commands can be written much more simply. Enter this example on one line:

```
(echo Data files %_date & dir *.dat %+ echo `` & echo Text files %_date
& dir *.txt) > filelist
```

The redirection, which appears outside the parentheses, applies to all the commands within the parentheses. Because the redirection is performed only once, the commands will run slightly faster than if each command was entered separately. The same approach can be used for input [redirection](#) and [piping](#).

You can also use command grouping in a batch file or at the prompt to split commands over several lines. This last example is like the redirection example above, but is entered at the prompt. Note the **More?** prompt after each incomplete line. None of the commands are executed until the command group is completed with the closing parenthesis. This example does not have to be entered on one line:

```
[c:\] (echo Data files %_date
More? dir *.dat
More? echo.
More? echo Text files %_date
More? dir *.txt) > filelist
[c:\]
```

## Limitations

A group of commands in parentheses is like a long command line. The total length of the group may not exceed 65,535 characters before alias and variable expansion, and 131,071 characters after expansion, whether the commands are entered from the prompt, an alias, or a batch file.

You cannot use [TEXT / ENDTEXT](#), or [GOTO](#) or [GOSUB](#) labels in a command group.

Each line you type at the normal prompt or the **More?** prompt, and each individual command within the line, must be within the usual [command line length limit](#).

#### 4.5.19 Starting Applications

**TCC** offers several ways to start applications.

First, you can simply type the name of any application at the prompt. As long as the application's executable file is in one of the standard search directories (see below), **TCC** will find it and start it. If you type the full path name of the executable file at the prompt the application will be started even if it is not in one of the standard search directories.

**TCC** offers two methods to simplify and speed up access to your applications. One is to create an [alias](#), for example:

```
alias myapp d:\apps\myapp.exe
```

In **Take Command** you can also use the Tool Bar to start frequently used applications. For example, a tool bar button named **MyApp** which invokes the command *d:\apps\myapp.exe* would accomplish the same thing as the alias shown above. You can use these methods together. For example, if you define the alias shown above you can set up a tool bar button called **MyApp** and simply use the command **myapp** for the button, which would then invoke the previously-defined alias.

You can also start an application by typing the name of a data file associated with the application. **TCC** will examine the file's extension and run the appropriate application, based on [executable extensions](#) or [Windows file associations](#).

For additional flexibility, you can also start applications with the [START](#) command. [START](#) provides a number of switches to customize the way an application is started.

#### Searching for Applications

When you start an application without specifying a path, **TCC** searches for the application in the current directory, and then all directories on the PATH. **TCC** also searches the **Windows** and **Windows system** directories; see the [PATH](#) command for details. (If you do enter an explicit path, **TCC** will only look in the directory you specified.)

If you enter a file name with no extension, **TCC** will search each directory for a matching .EXE, .BTM, .BAT, or .CMD file (and .REXX and/or .REXX if a REXX interpreter is loaded), then for a file matching a Windows file association or executable extension. That search order may be altered via the [PathExt](#) configuration option. If no such file is found, **Take Command** will move on to the next directory in the search sequence.

#### Take Command Application Windows

**Take Command** runs console (character mode) applications either in a tab window within **Take Command** or in their own console window. **Take Command** usually starts GUI applications in their own window, but you can also run simple GUI apps in a tab window (provided the application does not have multiple parent windows) with the [Run](#) dialog or the [START/TAB](#) option.

#### 4.5.20 Waiting for Applications to Finish

When you start a Windows GUI application from the prompt, **TCC** does not normally wait for the application to finish before returning to the prompt. This allows you to continue your work at the prompt while the application is running. You can force **TCC** to wait for applications to finish before continuing by selecting the [Wait for External Apps](#) configuration option, or with the [START](#) command's /WAIT switch ([START](#) can also control many other aspects of how your applications are started).

**TCC** always waits for applications that are run from transient shells (with a /C), or from batch files before continuing with subsequent commands in the batch file. To start an application from a transient shell or a batch file and continue without waiting for the application to finish, use the [START](#) command (without the /WAIT switch).

Due to the way Windows handles URLs, you cannot wait for the browser to finish when you enter an HTTP: URL at the prompt. In this situation, **TCC** always displays the next prompt immediately.

#### 4.5.21 Escape Character

**TCC** recognizes a user-definable escape character. This character gives the character that follows a special meaning; it has a different purpose than the ASCII **ESC** that is often used in ANSI X3.64 and printer control sequences.

The default escape character is a caret (^, ASCII: 94). If you don't like the default escape character, you can pick another character using the [SETDOS /E](#) command, or the [Escape character](#) configuration option. If you plan to share aliases or batch files on other **TCC** configurations, use the [%=\\_](#) pseudovariable, which is accepted in all of them, regardless of the actual value assigned to the escape character. See the section on [Special Character Compatibility](#) for details about choosing compatible escape characters. Note that if you change the default, your batch files will not work under CMD and you won't be able to run third-party batch files.

Ten special characters are recognized when they are preceded by the escape character. The combination of the escape character and one of these characters is translated to a single character, as shown below. The special characters which can follow the escape character are:

##### **Codes for Escape Characters**

- b** backspace
- c** comma ,
- e** the ASCII **ESC** character (code 27)
- f** form feed
- k** back quote `
- n** line feed
- q** double quote "
- r** carriage return
- s** space
- t** horizontal tab character

If you follow the escape character with any other character, the escape character is removed and the second character is copied directly into the command line. This allows you to suppress the normal meaning of special characters (such as ? \* \ | " ` > < and &). For example, to display a message containing a > symbol, which normally indicates redirection:

```
echo 2 is ^> 4
```

The escape character has an additional use when it is the last character on any line of a batch file. **TCC** recognizes this use of the escape character to signal line continuation: it removes the escape character and appends the next line to the current line before executing it.

**WARNING:** Escape characters are considered to be normal characters on the right side of a pipe.

**Note:** The term **escape character** has two additional usages not related to the above description, as detailed in the description of the [PROMPT](#) command and in [ASCII, Key Codes and Key Names](#).

## 4.5.22 Command Parsing

Whenever you type something at the command line and press the [Enter](#) key, or include a command in a batch file, you pass a command to **TCC**, which must determine how to execute it. If you understand the general process that is used, you will be able to make the best use of the commands. Understanding these steps can be especially helpful when working with complex aliases or batch file commands.

**TCC** goes through several steps when parsing a command line. Before it starts, it writes the entire command line (which may contain [multiple commands](#)) to the history log file if history logging has been enabled (with the [LOG /H](#) command) and the command did not come from a batch file. The first command is then isolated for processing. The following steps outline the basic processing required for each command. During that processing, additional parsing tasks may be triggered as noted and some steps may be repeated multiple times.

### 1. Separating the command from its tail

**TCC** begins by dividing the command into a command name and a command tail. The command name is the first word in the command, and the tail is everything that follows the command name. For example, in the command line

```
dir *.txt /2/p/v
```

The command name is **dir**, and the command tail is "**\*.txt /2/p/v**". In some instances, the parser will be able to understand incorrect syntax such as **dir/w**, but there should always be at least one space between the command name and its parameters.

### 2. Expanding aliases

Next, **TCC** tries to match the command name against its list of [aliases](#). If it finds a match between the command name and one of the aliases you've defined, it replaces the command name with the contents of the alias. This substitution is done internally and is not normally visible to you; however, you can view a command line with aliases expanded by pressing [Ctrl-F](#) after entering the command at the prompt.

If the alias included parameters (%1, %2, etc.), the parameter values are filled in from the text on the command line, and any parameters used in this process are removed from the command line. The process of replacing a command name that refers to an alias with the contents of the alias, and filling in the alias parameters, is called alias expansion.

This expansion of an alias creates a new command name: the first word of the alias. This new command name is again tested against the list of aliases, and if a match is found the contents of the new alias is expanded just like the first alias. This process, called nested alias expansion, continues until the command name no longer refers to an alias.

### 3. Expanding variables

The next step is to locate any batch file parameters, environment variables, internal variables, or variable functions in the command, and replace each one with its value (see "[Environment: Variables and Functions](#)"). This process is called variable expansion, and is not normally visible. However, you can view an expanded command line by pressing [Ctrl-X](#) after entering the command at the prompt.

The variable expansion process is modified for certain internal commands, such as [EXCEPT](#), [IF](#), and [GLOBAL](#). These commands are always followed by another command, so variable expansion takes place separately for the original command and the command that follows it.

### 4. Identifying a plugin or internal command

Once it has finished variable expansion, **TCC** next tries to match the resulting command name with its list of [plugin](#) commands or [internal commands](#). If it is unsuccessful, it knows that it will have to search for a batch file or external program to execute your command.

### 5. Displaying the command

When all of the aliases and environment variables have been expanded, **TCC** will echo the complete command to the screen (if command line echo has been enabled) and write it to the log file (if command [logging](#) has been turned on).

### 6. Processing redirection and piping

Before it can actually execute your command, **TCC** must scan the command tail to see if it includes [redirection](#) or [piping](#). If so, the proper internal switches are set to send output to an alternate device or to a file instead of to the screen. A second process is started at this point, if necessary, to receive any piped output.

### 7. Processing escape characters

At this stage, any remaining [Escape Characters](#) are processed. However, this might also already have taken place inside some of the variable functions (such as [@IF](#)) that are likely to pass escaped strings in their parameters. If you are referencing one of those in an [ECHO](#) or similar command, you need to escape twice ("^") or use [SETDOS /X](#) to avoid premature evaluation. Carefully test those situations to make sure the results are as you intended.

### 8. Executing the command

Finally, it is time to execute the command. **TCC** will first look for a matching [plugin](#) command name; if it doesn't exist then it tries to match an internal command. Otherwise, **TCC** searches for an executable (.EXE) file, a batch file, or a file with an executable extension that matches the command name (see the detailed description of this search in [Executable Files and File Searches](#)).

### 9. Cleaning up

Once the internal command or external program has terminated, **TCC** saves the result or exit code that the command generated, cleans up any redirection that you specified, and then returns to the original command line to retrieve the next command. When all of the commands in a command line are finished, the next line is read from the current batch file, or if no batch file is active, the prompt is displayed.

**Note:** You can disable and reenable several parts of command parsing (for example alias expansion, variable expansion, and redirection) with the [SETDOS /X](#) command.

### 4.5.23 Command Line Length Limits

When you first enter a command at the command prompt, in an alias or function definition, or in a batch file, it can be up to 65,535 characters long.

As **TCC** scans the command line and substitutes the contents of aliases user defined functions, and environment variables for their names, the line usually gets longer. This expanded line is limited to 131,071 characters. If your use of aliases, user defined functions, or environment variables causes the command line to exceed the applicable one of these limits as it is expanded, you will see a **Command line too long** error and the remainder of the line will not be executed.

### 4.5.24 Special Character Compatibility

If you want to share aliases, user defined functions, and batch files with other users, you need to be aware of possible differences in three important characters: the Command Separator (see [Multiple Commands](#)), the Escape Character (see [Escape Character](#)), and the Parameter Character (see [Batch File Parameters](#)).

The default values of each of these characters is shown in the following chart.

| Product                | Separator | Escape | Parameter |
|------------------------|-----------|--------|-----------|
| <b>4DOS</b> (obsolete) | ^         | Ctrl-X | &         |
| <b>TCC</b> (and CMD)   | &         | ^      | \$        |
| <i>pseudovariable</i>  | %+        | %=     |           |

In your batch files and aliases, and even at the command line, you can smooth over these differences in three ways:

1. Use internal pseudovariables that contain the current special character, rather than using the character itself (see [%+](#) and [%=](#)). For example, this command:

```
if "%1" == "" (echo Parameter missing! & quit)
```

will only work if the command separator is an ampersand. However, this version works regardless of the current command separator:

```
if "%1" == "" (echo Parameter missing! %+ quit)
```

2. Select a consistent set of characters from the [Advanced tab](#) of the [configuration dialogs](#).
3. In a batch file, use the [SETLOCAL](#) command to save the command separator, escape character, and parameter character when the batch file starts. Then use [SETDOS](#) as described below to select the characters you want to use within the batch file. Use an [ENDLOCAL](#) command at the end of the batch file to restore the previous settings.

You can also use the [SETDOS](#) command to change special characters on the command line.

## 4.5.25 Date Input Formats

### Date Input Formats

Commands and functions which accept a date as a parameter expect the same field order displayed by the [DIR](#) command and functions returning a date without a format code specifier. The year can be entered as a 4-digit or 2-digit value. Two-digit years from 80 to 99 are interpreted as 1980...1999; values from 0 to 79 are interpreted as 2000...2079. Month and day may be entered without a leading zero. Most non-numeric printing characters are accepted as field separators. All three fields must be specified, except for the ISO day format (yyyy-ddd) which requires two fields.

## 4.5.26 Case Sensitivity

With the following exceptions, **TCC** treats upper case and lower case letters identically:

The relational operator **EQC** (in IF, IFF, DO, etc.)

The character manipulation functions [@ascii](#), [@unicode](#), [@repeat](#), [@replace](#), [@similar](#), [@strip](#) and [@wild](#).

The codes used to specify units of storage size (**kKmMgGtT**) in:

- size ranges
- disk space and file size reporting functions

## 4.5.27 Directory Navigation

**TCC** remembers both a current or default drive for your system as a whole, and a current or default directory for every drive in your system. The current directory on the current drive is sometimes called the current working directory.

With traditional command processors, you change the current drive by typing the new drive letter plus a colon at the prompt. You change the current working directory with the [CD](#) command. **TCC** supports these standard features, and offer a number of enhancements to make directory navigation much simpler and faster.

This section begins with a summary of all the **TCC** directory navigation features. It also provides detailed documentation on the enhanced directory search features: [Extended Directory Searches](#) and [CDPATH](#).

The **TCC** directory navigation features are in three groups: features which help **TCC** find the directory you want, methods for initiating a directory change with a minimal amount of typing, and methods for returning easily to directories you've recently used. Each group is summarized below.

### Finding Directories

Traditional command processors require you to explicitly type the name of the directory you want to change to. **TCC** supports this method, and also offers two significant enhancements:

- ▶ The [CDPATH](#) variable allows you to enter a specific list of directories to be searched, rather than searching a database. Use [CDPATH](#) instead of Extended Directory Searches if you find the extended searches too broad, or your hard drive has too many directories for an efficient search.
- ▶ [Extended Directory Searches](#) allows **TCC** to search a database of all the directories on your system to find the one you want.

## Changing Directories

TCC supports the traditional methods of changing directories, and also offers several more flexible approaches:

- ▶ [Automatic directory changes](#) allow you to type a directory name at the prompt and switch to it automatically, without typing an explicit [CD](#) or similar command.
- ▶ The [CD](#) command can change directories on a single drive, and can return to the most recently used directory.
- ▶ The [CDD](#) command changes drive and directory at the same time, and can return to the most recently used drive and directory.
- ▶ The [PUSHD](#) command changes the drive and directory like [CDD](#), and records the previous directory in a directory "stack." You can view the stack with the [DIRS](#) command or the [@DIRSTACK](#) function, and return to the directory on the top of the stack with [POPD](#).

[CDD](#), [PUSHD](#), and [automatic directory changes](#) can also change to network drives and directories mapped to drive letters and to ones specified with UNC names (see [File Systems](#) for details).

## Returning to a Previous Directory

CMD does not remember previously-used directories, and can only "return" to a directory by changing back to it with a standard drive change or CD command. TCC supports three additional, simpler methods for returning to a previous directory:

- ▶ The [CD -](#) and [CDD -](#) commands can be used to return to the previous working directory (the one you used immediately before the current directory). Use these commands if you are working in two directories and alternating between them.
- ▶ The [directory history window](#) allows you to select one of several recently-used directories from a popup list and return to it immediately. The window displays the contents of the directory history list.
- ▶ The [POPD](#) command returns to the last directory saved by [PUSHD](#). The directory stack holds 2048 characters, enough for 40 to 80 typical drive and directory entries.

### 4.5.27.1 CDPATH

When you change directories with an [automatic directory change](#) or the [CD](#), [CDD](#), or [PUSHD](#) command, TCC must find the directory you want to change to. If it cannot find an exact match of the directory path and name, TCC tries to find the directory you requested via the [CDPATH](#), then via an [Extended Directory Search](#).

Enabling both [CDPATH](#) and [Extended Directory Searches](#) can yield confusing results. If you prefer to explicitly specify where TCC should look for directories, use [CDPATH](#). If you prefer to have TCC look at all of the directory names on your disk, use Extended Directory Searches.

[CDPATH](#) is an environment variable, and is similar to the [PATH](#) variable used to search for executable files: it contains an explicit list of directories to search when attempting to find a new directory. TCC appends the specified directory name to each directory in [CDPATH](#) and attempts to change to that drive and directory. It stops when it finds a match or when it reaches the end of the [CDPATH](#) list.

[CDPATH](#) is ignored if a complete directory name (one beginning with a backslash \) is specified, or if a drive letter is included in the name. It is only used when a name is given with neither drive letter nor leading backslash.

[CDPATH](#) provides a quick way to find commonly used subdirectories in an explicit list of locations. You can create [CDPATH](#) with the [SET](#) command. The format of [CDPATH](#) is similar to that of [PATH](#): a list of directories separated by semicolons. For example, if you want the directory change commands to search the *C:\DATA* directory, the *D:\SOFTWARE* directory, and the root directory of drive *E*: for the subdirectories that you name, you should create [CDPATH](#) with this command:

```
set cdpwd=c:\data;d:\software;e:\
```

Suppose you are currently in the directory *C:\WP\LETTERS\JANUARY*, and you'd like to change to *D:\SOFTWARE\UTIL*. You could change directories explicitly with the command:

```
[c:\wp\letters\january] cdd d:\software\util
```

However, because the *D:\SOFTWARE* directory is listed in your [CDPATH](#) variable as shown in the previous example (we'll assume it is the first directory in the list with a *UTIL* subdirectory), you can simply enter the command

```
[c:\wp\letters\january] cdd util
```

or, using an automatic directory change:

```
[c:\wp\letters\january] util\
```

to change to *D:\SOFTWARE\UTIL*.

**TCC** looks first in the current directory, and attempts to find the *C:\WP\LETTERS\JANUARY\UTIL* subdirectory. Then it looks at [CDPATH](#), and appends *UTIL* to each entry in the [CDPATH](#) variable. In other words, it tries to change to *C:\DATA\UTIL*, then to *D:\SOFTWARE\UTIL*. Because this change succeeds, the search stops and the directory change is complete.

If you often switch between "sibling" directories, i.e., between subdirectories of a common parent directory, you can enter .. as a search entry in your [CDPATH](#). You can use ... to find "uncles", i.e., a directory one level up (a sibling of the parent directory), thus a subdirectory of the directory 2 levels up.

#### 4.5.27.2 Extended Directory Searches

When you change directories with an [automatic directory change](#), [CD](#), [CDD](#), or [PUSHD](#) command, **TCC** must find the directory you want to change to. To do so, it first checks to see whether you have specified either the name of an existing subdirectory below the current directory, or the name of an existing directory with a relative or full path or a drive letter. If you have, **TCC** changes to that directory, and does no further searching.

This search method requires that you navigate manually through the directory tree, and type the entire name of each directory you want to change to. Extended Directory Searches speed up the navigation process dramatically by allowing **TCC** to find the directory you want, even if you only enter a small part of its name.

When the first search method fails, **TCC** tries to find the directory you requested via the [CDPATH](#) variable, then via an Extended Directory Search. This section covers only Extended Directory Searches, which are more flexible and more commonly used than [CDPATH](#).

Extended Directory Searches use a database of directory names to facilitate changing to the correct directory. The database is used only if Extended Directory Searches are enabled, and if the explicit directory search and [CDPATH](#) search fail to find the directory you requested.

An extended directory search automatically finds the correct path to the requested directory and changes to it if that directory exists in your directory database. If more than one directory in the database matches the name you have typed, a popup window appears and you can choose the directory you want.

If the TCMD.INI directive [EverythingSearch](#) is set, **TCC** will use **Everything Search** (free at [http://www voidtools.com](http://www voidtools com)) instead of its own database for fuzzy directory searches. **Everything Search** is slightly faster, but will only work on local NTFS drives. Setting EverythingSearch is the equivalent of setting FuzzyCD=3 (\*name\*). You must download and install **Everything Search** yourself; it is not included in the **Take Command** distribution.

You can move and/or resize the directory search window. **TCC** will use the new position and size the next time the directory search window is invoked. You can also change the keys used in the popup window with [key mapping directives](#).

To use extended directory searches, you must explicitly enable them (see below) and also create the directory database.

### The Extended Search Database

To create or update the database of directory names, use the [CDD /S](#) command. When you create the database with CDD /S, you can specify which drives should be included. If you enable Extended Directory Searches and do not create the database, it will be created automatically the first time it is required, and will include all local hard drives.

The database is stored in the file *JPSTREE.IDX*. By default, the file is placed in the root directory of drive C:. Because of security restrictions in Windows Vista and later, the the default directory is defined as the value of the environment variable LOCALAPPDATA (predefined by Windows). If you are running Vista, Windows 7 or Windows 8 and don't have LOCALAPPDATA in your environment, the default directory will be the directory where **TCC** is installed. You can specify a different location for this file on the [Command Line tab](#) of the [configuration dialogs](#).

If you use an internal command to create or delete a directory, the directory database is automatically updated to reflect the change to your directory structure.

The [TREEEXCLUDE](#) variable can be used to specify which drives/directories should be excluded from inclusion in the directory database.

The internal commands which can modify the directory structure and cause automatic updates of the file are [MD](#), [RD](#), [COPY /S](#), [DEL /X](#), [MOVE /S](#), and [REN](#). The [MD /N](#) command can be used to create a directory without updating the directory database. This is useful when creating a temporary directory which you do not want to appear in the database.

### Enabling Extended Searches

To enable extended directory searches and control their operation, you must set the Search Level on the [Command Line tab](#) of the [configuration dialogs](#).

- If Search Level = 0, extended searches are disabled, the *JPSTREE.IDX* database is ignored, and [CD](#), [CDD](#), [PUSHD](#) and automatic directory changes search for directories using only explicit names and [CDPATH](#). This is the default.
- If Search Level = 1 and an extended search is required, **TCC** will search the *JPSTREE.IDX* database for directory names which exactly match the name you specified.
- If Search Level = 2 and an extended search is required, **TCC** will search the database for exact matches first, just as when Search Level = 1. If the requested directory is not found, it will search the database a second time looking for directory names that begin with the name you specified.
- If Search Level = 3 and an extended search is required, **TCC** will search the database for exact matches first, just as when Search Level = 1. If the requested directory is not found, it will search the database a second time looking for directory names that contain the name you specified anywhere within them.

For example, suppose that you have a directory called *C:\DATA\MYDIR*, [CDPATH](#) is not set, and *C:\DATA* is not the current directory on drive C:. The following chart shows what [CDD](#) command you might use to change to this directory.

| Search Level | Type of extended search                 | Typical CDD Command |
|--------------|-----------------------------------------|---------------------|
| 0            | <a href="#">CDPATH</a> only (default)   | cdd c:\data\mydir   |
| 1            | <a href="#">CDPATH</a> or exact match   | cdd mydir           |
| 2            | <a href="#">CDPATH</a> or leading match | cdd myd             |
| 3            | <a href="#">CDPATH</a> or any match     | cdd yd              |

An extended directory search is not used if you specify a full directory path (one beginning with a backslash \, or a drive letter and a backslash). If you use a name which begins with a drive letter (e.g. *C:MYDIR*), the extended search will examine only directories on that drive.

### Forcing an Extended Search with Wildcards

Normally you type a specific directory name for **TCC** to locate, and the search proceeds as described in the preceding sections. However, you can also force **TCC** to perform an extended directory search by using [wildcard characters](#) in the directory name. If you use a wildcard, an extended search will occur whether or not extended searches have been enabled.

When **TCC** is changing directories and it finds *wildcards* in the directory name, it skips the explicit search and [CDPATH](#) steps and goes directly to the extended search.

If a single match is found, the change is made immediately. If more than one match is found, a popup window is displayed with all matching directories.

Wildcards can only be used in the final directory name in the path (after the last backslash in the path name). For example you can find *COMM\*A\** (all directories whose parent directory is *COMM* and which have an **A** somewhere in their names), but you cannot find *CO?M\ \*A\** because it uses a wildcard before the last backslash.

If you use wildcards in the directory name as described here, and the extended directory search database does not exist, it will be built automatically the first time a wildcard is used. You can update

the database at any time with [CDD /s](#).

Internally, extended directory searches use wildcards to scan the directory database. If Search Level is set to 2, an extended search looks for the name you typed followed by an asterisk (i.e. *DIRNAME\**). If Search Level is set to 3, it looks for the name preceded and followed by an asterisk (i.e. *\*DIRNAME\**).

These internal wildcards will be used in addition to any wildcards you use in the name. For example if you search for *ABC?DEF* (*ABC* followed by any character followed by *DEF*) and Search Level is set to 3, **TCC** will search the directory database for *\*ABC?DEF\**.

### Disabling Extended Searches in Batch Files

When writing batch files you may want to use the [CD](#) or [CDD](#) command to switch directories without triggering an extended search. For example, you may need the search to fail (rather than search the extended search database) if a directory does not exist, or you may want to ensure that the extended search popup window does not appear in a batch file designed to run in unattended mode.

To disable extended searches, use the /N option of [CD](#) or [CDD](#). When this option is used and a directory does not exist below the current directory or on the [CDPATH](#), the command will fail with an error message, and will not search the extended search database. For example this command might trigger an extended search:

```
cdd testdir
```

but this one will not:

```
cdd /n testdir
```

Note that this option is not available for [PUSHD](#). To perform the same function when using [PUSHD](#), save the current directory with [PUSHD](#) (without parameters) and then use [CDD /N](#) to change directories, for example:

```
pushd
cdd /n testdir
```

#### 4.5.27.3 Automatic Directory Changes

Automatic directory changes are part of the comprehensive directory navigation features built into **TCC**. For a summary of these features, and more information on [Extended Directory Searches](#) and [CDPATH](#), see [Directory Navigation](#).

Automatic directory changes let you change directories quickly from the command prompt, without entering an explicit [CD](#) or [CDD](#) command. Simply type the name of the directory you want to change to at the prompt, with a terminating backslash (\) (either entered manually, or automatically via the [Add \ to Directories](#) configuration option). For example:

```
[c:\] tcmd\
[c:\tcmd]
```

This can make directory changes very simple when it's combined with [Extended Directory Searches](#) or [CDPATH](#). If you have enabled either of those features, **TCC** will use them in searching for a directory with an automatic directory change.

For example, suppose [Extended Directory Searches](#) are enabled, and the directory *WIN* exists on drive

*E*:: You can change to this directory with a single word on the command line:

```
[c:\tcmd] win\
[e:\win]
```

This depends on the way Extended Directory Changes are configured, and the number of subdirectories on your disk whose names contain the string **WIN**, when you execute such a command you may see an immediate change as shown above, or a popup window which contains a list of subdirectories matching **WIN** to choose from.

The text before the backslash can include a drive letter, a full path, a partial path, or a UNC name (see [File Systems](#) for details on UNC names). Commands like "...\" can be used to move up the directory tree quickly (see [Extended Parent Directory Names](#)).

If you enter a directory name without the trailing backslash, the parser will change to that directory if no internal or external command of that name is found (and before the [UNKNOWN\\_CMD](#) alias is executed.)

All directory changes, including automatic ones, save the current directory so it can be recalled with a [CDD](#) - or [CD](#) - command.

For example, any of the following are valid automatic directory change entries:

```
[c:\] d:\data\finance\
[c:\] archives\
[c:\] ...util\scanner\
[c:\] \\server\vol1\george\
```

The first and last examples change to the named directory. The second changes to the *ARCHIVES* subdirectory of the current directory, and the third changes to the *UTIL\SCANNER* subdirectory of the directory which is two levels up from the current directory in the tree.

#### 4.5.27.4 Directory Aliases

Directory Aliases are a shorthand way of specifying pathnames. For example, if you define an alias:

```
alias pf:=c:\program files
```

You can then reference the files in **c:\program files\jpsoft** by entering **pf:jpsoft**. Directory aliases work in places that accept filenames and directory names (internal command arguments or the first argument in a command line), including filename completion. You cannot use them in arguments to external applications, as **TCC** has no way of knowing what is a valid argument for external applications.

Directory alias names can be either two or more alphanumeric characters followed by a colon, or a single digit followed by a colon.

Directory aliases support environment variable expansion.

## 4.6 Aliases & Batch Files

Whenever you have a command (internal or external) that you need to execute often, one that's too complex to be dependably typed manually at the [Command Line](#), one that needs to be part of an exact sequence of other commands, one that you want to be able to easily repeat from another location or share with others, or you repeat very often and therefore want to have a very short name, you can store that command as part of a convenient ALIAS and/or batch file.

- ▶ [Aliases](#)
- ▶ [Batch Files](#)
- ▶ [Special Character Compatibility](#)

#### 4.6.1 Aliases

Much of the power of **TCC** comes together in **aliases**, which give you the ability to create your own commands. An alias is a name that you select for a command or group of commands. Simple aliases substitute a new name for an existing command. More complex aliases can redefine the default settings of internal or external commands, operate as very fast in-memory batch files, and perform commands based on the results of other commands. **TCC** also supports [Directory Aliases](#), a shorthand way of specifying pathnames. **TCC** supports either a local alias list that is only visible to the current **TCC** session, or a global alias list that is shared among all **TCC** sessions.

This section shows you some examples of the power of aliases. See the [ALIAS](#) command for complete details about writing your own aliases.

The simplest type of alias gives a new name to an existing command. For example, you could create a command called **R** (for Root directory) to switch to the root directory this way:

```
alias r=cd \
```

After the alias has been defined this way, every time you type the command **R**, you will actually execute the command [CD](#) \.

Aliases can also create customized versions of commands. For example, the [DIR](#) command can sort a directory in various ways. You can create an alias called **DE** that means "sort the directory by filename extension, and pause after each page while displaying it" like this:

```
alias de=dir /oe /p
```

Aliases can be used to execute sequences of commands as well. The following command creates an alias called **MUSIC** which saves the current drive and directory, changes to the **SOUNDS** directory on drive **c**, runs the program **E:\MUSIC\PLAYER.EXE**, and, when the program terminates, returns to the original drive and directory (enter this on one line):

```
alias music='pushd c:\sounds & e:\music\player.exe & popd'
```

This alias is enclosed in back-quotes because it contains multiple commands. You must use the back-quotes whenever an alias contains multiple commands, environment variables, parameters (see below), redirection, or piping. See the [ALIAS](#) command for full details.

When an alias contains multiple commands, the commands are executed one after the other. However, if any of the commands runs an external Windows application (such as the fictitious **PLAYER.EXE** shown above), you must be sure the alias will wait for the application to finish before continuing with the other commands. See [Waiting for Applications to Finish](#) for additional details.

Aliases can be nested; that is, one alias can invoke another. For example, the alias above could also be written as:

```
alias play=e:\music\player.exe
alias music='pushd c:\sounds & play & popd'
```

If you enter *MUSIC* as a command, **TCC** executes the [PUSHD](#) command, detects that the next command (**PLAY**) is another alias and executes the program *E:\MUSIC\PLAYER.EXE*, and, when that program exits, returns to the first alias, executes the [POPD](#) command, and returns to the prompt.

You can use aliases to change the default options for both internal commands and external commands. Suppose that you always want the [DEL](#) command to prompt before it erases a file:

```
alias del=*del /p
```

An asterisk \* is used in front of the second [DEL](#) to tell **TCC** to use the original internal command, not an alias. See [Temporarily Disabling Aliases](#) for more information about this use of the asterisk.

You may have a program on your system that has the same name as an internal command. Normally, if you type the command name, you will start the internal command rather than the program you desire, unless you explicitly add the program's full path on the command line. For example, if you have a program named *DESCRIBE.EXE* in the **C:\WUTIL** directory, you could run it with the command *C:\WUTIL\DESCRIBE.EXE*. However, if you simply type *DESCRIBE*, the internal [DESCRIBE](#) command will be executed instead. Aliases give you two simple ways to get around this problem.

First, you could define an alias that runs the program in question, but using a different name:

```
alias desc=c:\winutil\describe.exe
```

Another approach is to use an alias to rename the internal command and use its original name for the external program. The following example creates the alias [FILEDESC](#) for the [DESCRIBE](#) command, and then uses a second alias to run *DESCRIBE.EXE* whenever you type *DESCRIBE*:

```
alias filedesc=*describe
alias describe=c:\winutil\describe.exe
```

You can also assign an alias to a key, so that every time you press the key, the command will be invoked. You do so by naming the alias with an at sign [@] followed by a key name. After you enter this next example, you will see a 2-column directory with paging whenever you press **Shift-F5** followed by **Enter**:

```
alias @Shift-F5=*dir /2/p
```

This alias will put the [DIR](#) command on the command line when you press **Shift-F5**, then wait for you to enter file names or additional switches. You must press Enter when you are ready to execute the command. To execute the command immediately, neither displaying it on the command line, nor waiting for you to press Enter, use two @ signs at the start of the alias name:

```
alias @@Shift-F5=*dir /2/p
```

The next example clears the window whenever you press **Ctrl-F2**:

```
alias @@Ctrl-F2=cls
```

Aliases have many other capabilities as well. The next example creates a simple command line calculator. Once you have entered the example, you can type **CALC 4\*19**, for example, and you will see the answer:

```
alias calc=`echo The answer is: %@eval[%$]`
```

Our last example in this section creates an alias called **IN**. It temporarily changes directories, runs an internal or external command, and then returns to the current directory when that command is finished:

```
alias in=`pushd %1 & %2$ & popd`
```

Now if you type:

```
in c:\sounds play furelise.wav
```

you will change to the C:\SOUNDS subdirectory, execute the command **PLAY FURELISE.WAV**, and then return to the current directory.

### Alias Parameters

The above example uses two parameters: **%1** means the first parameter on the command line, and **%2\$** means the second and all subsequent parameters.

Aliases can use command line parameters or parameters like those in batch files. The command line parameters are numbered from **%0** to **%511**. (**%0** contains the alias name.) You can use double quotes to pass spaces, tabs, commas, and other special characters in an alias parameter; see [Parameter Quoting](#) for details. Alias examples in this section assume the **TCC** default of ParameterChar=\$.

Parameters that are referred to in an alias, but which are missing on the command line, appear as empty strings inside the alias. For example, if you only put two parameters on the command line, any reference in the alias to **%3** or any higher-numbered parameter will be interpreted as an empty string.

The parameter **%n\$** has a special meaning. **TCC** interprets it to mean "the entire command line, from parameter **n** to the end." If **n** is not specified, it has a default value of **1**, so **%\$** means "the entire command line after the alias name."

The parameter **%-n\$** means "the command line from parameter 1 to **n - 1**".

The special parameter **%#** contains the number of command line parameters.

Aliases cannot use indirect access to command parameters, e.g., **%[%n]** (where **n** is a parameter number) does not return the selected parameter.

See the [ALIAS](#) and [UNALIAS](#) commands for more information and examples.

## 4.6.2 Batch Files

A batch file is a file that contains a list of commands to execute. **TCC** reads and interprets each line as if it had been typed at the keyboard. Like [aliases](#), batch files are handy for automating computing tasks. Unlike aliases, batch files can be as long as you wish. Batch files take up separate disk space for each file, and can't usually execute quite as quickly as aliases, since they must be read from the disk.

Some of the topics included in this section are:

- ▶ [.BAT, .CMD, and .BTM](#)
- ▶ [Echoing in Batch Files](#)
- ▶ [Batch File Line Continuation](#)

- [Batch File Parameters](#)
- [Using Environment Variables](#)
- [Batch File Commands](#)
- [Interrupting a Batch File](#)
- [Automatic Batch Files](#)
- [Detecting TCC and Take Command](#)
- [Using Aliases in Batch Files](#)
- [Debugging Batch Files](#)
- [String Processing](#)
- [Batch File](#)
- [Lua Support](#)
- [Perl Support](#)
- [Perl Support](#)
- [Python Support](#)
- [REXX Support](#)
- [Ruby Support](#)
- [Tcl/tk Support](#)
- [EXTPROC / Shebang Support](#)

#### 4.6.2.1 .BAT, .CMD & .BTM Files

A batch file can run in two different modes. In the first, traditional mode, each line of the batch file is read and executed individually, and the file is opened and closed to read each line. In the second mode the batch file is opened once, the entire file is read into memory, and the file is closed. Only the first mode can be used for self-modifying batch files (which are rare).

The batch file's extension determines its initial mode. Files with a *.BAT* or *.CMD* extension are run in the first mode. Files with a *.BTM* extension are run in the more efficient second mode. You can change the execution mode inside a batch file with the [LOADBTM](#) command.

#### 4.6.2.2 Echoing in Batch Files

By default, each line in a batch file is displayed or "echoed" as it is executed. You can change this behavior, if you want, in several different ways:

- Any batch file line that begins with an @ symbol will not be displayed.
- The display can be turned off and on within a batch file with the [ECHO OFF](#) and [ECHO ON](#) commands.
- The default setting can be changed with the [SETDOS /V](#) command, or the [Default Batch Echo](#) configuration option.

For example, the following line turns off echoing inside a batch file. The @ symbol keeps the batch file from displaying the ECHO OFF command itself:

```
@echo off
```

TCC also has a command line echo that is unrelated to the batch file echo setting. See [ECHO](#) for details about both settings.

#### 4.6.2.3 Special syntax for CMD compatibility

For compatibility with CMD, TCC supports additional syntax to qualify references to parameters of batch files and the control variable of the [FOR](#) command when referenced by the **command** it executes.

However, this syntax can usually be replaced by more flexible [Variable Functions](#).

| CMD syntax              | Expands to                                                                | Suggested replacement                       |
|-------------------------|---------------------------------------------------------------------------|---------------------------------------------|
| <code>%*</code>         | All parameters                                                            | <code>%\$</code>                            |
| <code>%~n</code>        | unquoted ("")                                                             | <code>%@replace[^",,%n]</code>              |
| <code>%~fn</code>       | Fully qualified name of <code>%n</code>                                   | <code>%@full[%n]</code>                     |
| <code>%~dn</code>       | Drive letter portion of <code>%n</code>                                   | <code>%@left[2,%@full[%n]]</code>           |
| <code>%~pn</code>       | Full path (no drive letter) of <code>%n</code>                            | <code>%@right[-2,%@path[%@full[%n]]]</code> |
| <code>%~nn</code>       | Root name (no extension) of <code>%n</code>                               | <code>%@name[%n]</code>                     |
| <code>%~xn</code>       | File extension of <code>%n</code>                                         | <code>.%@ext[%n]</code>                     |
| <code>%~sn</code>       | Fully qualified short name of <code>%n</code>                             | <code>%@sfn[%n]</code>                      |
| <code>%~an</code>       | File attributes of <code>%n</code>                                        | <code>%@attrib[%n]</code>                   |
| <code>%~tn</code>       | File date and time of <code>%n</code>                                     | <code>%@filedate[%n] %@filetime[%n]</code>  |
| <code>%~zn</code>       | File size of <code>%n</code> , bytes                                      | <code>%@filesize[%n]</code>                 |
| <code>%~\$PATH:n</code> | Full name of the first match for <code>%n</code> in <a href="#">%PATH</a> | <code>%@search[%n]</code>                   |

### Notes

In the special case where the parameter to a `%~` variable is **0**, e.g., `%~f0`, the returned file name will always include the extension, as it does under CMD.

`%~$PATH:n` returns an empty string if the file `%n` is not found in the path.

References qualified by the tilde ~ trigger an error message when used improperly, e.g. if attempting to display the size of a string parameter which is not the name of a file.

#### 4.6.2.4 Batch File Line Continuation

TCC will combine multiple lines in the batch file into a single line for processing when the [Escape Character](#) (the actual token or the symbolic "`%=`" reference) is the last character of each line to be combined (except the last). For example:

```
c:\> echo The quick brown fox jumped over the ^
sleeping ^
dog. > alphabet
```

You cannot use this technique to extend a batch file line beyond the normal [command line length limit](#).

#### 4.6.2.5 Batch File Parameters

Like [aliases](#), user-defined [functions](#) and application programs, batch files can examine the command line that is used to invoke them. The command tail (everything on the command line after the batch file or alias name) is separated into individual positional parameters (also called parameters or batch variables) by scanning for the spaces, tabs, commas, and equals signs (=) that separate them. (The = separator can be disabled by setting the "[CMDBatchDelimiters](#)=No" directive in your TCMD.INI.) For aliases and functions, a forward slash (/) triggers the beginning of a new parameter, e.g. the string `xyz/abc` is separated into parameters `foo` and `/abc`.

These parameters are numbered from %1 to %4095. %1 refers to the first parameter on the command line, %2 to the second, and so on. It is up to the batch file to determine the meaning of each parameter. You can use double quotes to pass spaces, tabs, commas, and other special characters in a batch file parameter; see [Parameter Quoting](#) for details.

Parameters that are referred to in a batch file, but which are missing on the command line, appear as empty strings inside the batch file. For example, if you start a batch file and put two parameters on the command line, any reference in the batch file to %3, or any higher-numbered parameter, will be interpreted as an empty string.

A batch file can use the special parameters shown in the table below:

| parameter | value                                                                                        |
|-----------|----------------------------------------------------------------------------------------------|
| %0        | the name of the batch file as entered on the command line                                    |
| %#        | the number of command line parameters, modified by <a href="#">SHIFT</a>                     |
| %n\$      | the command tail starting with parameter number <i>n</i> , modified by <a href="#">SHIFT</a> |
| %-n\$     | the command tail from parameter 1 to <i>n</i> - 1                                            |
| %\$       | the complete command tail, modified by <a href="#">SHIFT</a>                                 |
| %*        | the complete command tail, unmodified by <a href="#">SHIFT</a>                               |

For example, %3\$ means the third and all subsequent parameters. The values of %#, %n\$, %-n\$, and %\$ will change if you use the [SHIFT](#) command. To emulate CMD, [SHIFT](#) does not affect the value of %\*.

For example, if your batch file interprets the first parameter as a subdirectory name then the following line would move to the specified directory:

```
cd %1
```

A friendlier batch file would check to make sure the directory exists and take some special action if it doesn't:

```
iff isdir %1 then
 cd %1
else
 echo Subdirectory %1 does not exist!
 quit
endiff
```

(See the [IF](#) and [IFF](#) commands.)

Batch files can also use [environment variables](#), [internal variables](#), and [variable functions](#).

Batch file parameters may also use the special [CMD compatibility syntax](#).

#### 4.6.2.5.1 Parameter Quoting

As [TCC](#) parses the command line, it looks for the [command separator](#), [conditional commands](#) (|| and &&), white space (spaces, tabs, and commas), percent signs % which indicate [variables](#) or [batch file](#) parameters to be expanded, and [redirection and piping](#) characters >, <, and |.

Normally, these special characters cannot be passed to a command as part of a parameter. However, you can include any of the special characters in a parameter by enclosing the entire parameter in single back quotes ['] or double quotes ["]. Although both back quotes and double quotes will let you build

parameters that include special characters, they do not work the same way.

No alias or variable expansion is performed on a parameter enclosed in back quotes. Redirection symbols inside the back quotes are ignored. The back quotes are removed from the command line before the command is executed.

No alias expansion is performed when an expression is enclosed in double quotes. Redirection symbols inside double quotes are ignored. However, variable expansion **is** performed in expressions inside double quotes. The double quotes themselves will be passed to the command as part of the parameter.

For example, suppose you have a batch file *CHKNAME.BTM* which expects a name as its first parameter (%1). Normally the name is a single word. If you need to pass a two-word name with a space in it to this batch file you could use the command:

```
chkname `MY NAME`
```

Inside the batch file, %1 will have the value **MY NAME**, including the space. The back quotes caused **TCC** to pass the string to the batch file as a single parameter. The quotes keep characters together and reduce the number of parameters in the line.

For a more complex example, suppose the batch file *QUOTES.BAT* contains the following commands:

```
@echo off
echo Arg1 = %1
echo Arg2 = %2
echo Arg3 = %3
```

and that the environment variable FORVAR has been defined with this command:

```
set FORVAR=for
```

Now, if you enter the command

```
quotes `Now is the time %forvar` all good
```

The output from *QUOTES.BAT* will look like this:

```
Arg1 = Now is the time %forvar
Arg2 = all
Arg3 = good
```

But if you enter the command:

```
quotes "Now is the time %forvar" all good
```

The output from *QUOTES.BAT* will look like this:

```
Arg1 = "Now is the time for"
Arg2 = all
Arg3 = good
```

Notice that in both cases, the quotes keep characters together and reduce the number of parameters in the line.

The following example has 7 command line parameters, while the examples above only have 3:

```
quotes Now is the time %%forvar all good
```

(The double percent signs are needed in each case because the parameter is parsed twice, once when passed to the batch file and again in the ECHO command.)

When an alias is defined in a batch file or from the command line, its parameter can be enclosed in back quotes to prevent the expansion of replaceable parameters, variables, and multiple commands until the alias is invoked. See [ALIAS](#) for details.

You can disable and reenable back quotes and double quotes with the [SETDOS](#) /X command.

#### 4.6.2.6 Using Environment Variables

Batch files can use [environment variables](#), [internal variables](#), [variable functions](#), or [user-defined functions](#). You can use these variables and functions to determine system status (e.g., the CPU type), resource levels (e.g., the amount of free disk space), file information (e.g., the date and time a file was last modified), and other information (e.g., the current date and time). You can also perform arithmetic operations (including date and time arithmetic), manipulate strings and substrings, extract parts of a filename, and read and write files.

To create temporary variables for use inside a batch file, just use the [SET](#) command to store the information you want in an environment variable. Pick a variable name that isn't likely to be in use by some other program (for example, PATH would be a bad choice), and use the [UNSET](#) command to remove these variables from the environment at the end of your batch file. You can use [SETLOCAL](#) and [ENDLOCAL](#) to create a "local" environment so that the original environment will be restored when your batch file is finished.

Environment variables used in a batch file may contain either numbers or text. It is up to you to keep track of what's in each variable and use it appropriately; if you don't (for example, if you use [%@EVAL](#) to add a number to a text string), you'll get an error message or a meaningless return value.

#### 4.6.2.7 Batch File Commands

Some commands are particularly suited to batch file processing. Each command is explained in detail in the [Command Reference](#). Here is a list of some of the commands you might find most useful:

|                             |                                                                                                                                 |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <a href="#">ACTIVATE</a>    | activates another window                                                                                                        |
| <a href="#">BEEP</a>        | produces a sound of any pitch and duration through the computer's speaker                                                       |
| <a href="#">BREAKPOINT</a>  | set a breakpoint in the batch debugger                                                                                          |
| <a href="#">CALL</a>        | executes one batch file from within another                                                                                     |
| <a href="#">CANCEL</a>      | terminates all batch file processing                                                                                            |
| <a href="#">CLS</a>         | clears the <b>TCC</b> window                                                                                                    |
| <a href="#">COLOR</a>       | sets the <b>TCC</b> display colors                                                                                              |
| <a href="#">DEBUGSTRING</a> | send text to the debugger                                                                                                       |
| <a href="#">DEFER</a>       | defers a command until the batch file ends                                                                                      |
| <a href="#">DO</a>          | starts a loop. The loop can be based on a counter, or on a conditional expression, strings, or files. ENDDO terminates the loop |
| <a href="#">DRAWBOX</a>     | draws a box on the screen                                                                                                       |
| <a href="#">DRAWHLINE</a>   | draws horizontal lines on the screen                                                                                            |
| <a href="#">DRAWVLINE</a>   | draws vertical lines on the screen                                                                                              |
| <a href="#">ECHO</a>        | sends text to the standard output device                                                                                        |
| <a href="#">ECHOS</a>       | sends text to the standard output device                                                                                        |

|                           |                                                                                                                                           |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#">ECHOERR</a>   | sends text to the standard error device                                                                                                   |
| <a href="#">ECHOERR</a>   | sends text to the standard error device                                                                                                   |
| <a href="#">ENDLOCAL</a>  | restores the settings that were saved and allows specific variables to be exported (see <a href="#">SETLOCAL</a> )                        |
| <a href="#">ENDTEXT</a>   | ends the block of text started with <a href="#">TEXT</a>                                                                                  |
| <a href="#">EVENTLOG</a>  | writes a string to the Windows application event log                                                                                      |
| <a href="#">FOR</a>       | executes commands for each file that matches a set of wildcards, or each entry in a list                                                  |
| <a href="#">GOSUB</a>     | executes a subroutine inside a batch file (see <a href="#">RETURN</a> ).                                                                  |
| <a href="#">GOTO</a>      | branches to a different location in the batch file                                                                                        |
| <a href="#">IF</a>        | execute commands based on a conditional expression                                                                                        |
| <a href="#">IFF</a>       |                                                                                                                                           |
| <a href="#">INKEY</a>     | collects keyboard input and store it in environment variables                                                                             |
| <a href="#">INPUT</a>     | collects keyboard input and store it in environment variables                                                                             |
| <a href="#">JABBER</a>    | send an instant message (IM)                                                                                                              |
| <a href="#">KEYSTACK</a>  | sends keystrokes to applications                                                                                                          |
| <a href="#">LOADBTM</a>   | changes the batch file operating mode                                                                                                     |
| <a href="#">MSGBOX</a>    | displays a dialog box with standard buttons like Yes, No, OK, and Cancel, and returns the user's selection                                |
| <a href="#">ON</a>        | initializes error handling for Ctrl-C / Ctrl-Break, or for program and command errors                                                     |
| <a href="#">OSD</a>       | Display floating text on the desktop                                                                                                      |
| <a href="#">PAUSE</a>     | displays a message and waits for the user to press a key                                                                                  |
| <a href="#">PDIR</a>      | creates a customized DIR-like display of directory contents                                                                               |
| <a href="#">PLAYAVI</a>   | plays Windows .AVI files                                                                                                                  |
| <a href="#">PLAYSOUND</a> | plays Windows sound files                                                                                                                 |
| <a href="#">POSTMSG</a>   | send a message to a window                                                                                                                |
| <a href="#">QUERYBOX</a>  | displays a dialog box for text input                                                                                                      |
| <a href="#">QUIT</a>      | ends the current batch file and optionally returns an exit code                                                                           |
| <a href="#">REM</a>       | places a remark in a batch file                                                                                                           |
| <a href="#">RETURN</a>    | terminates a subroutine (see <a href="#">GOSUB</a> )                                                                                      |
| <a href="#">SCREEN</a>    | positions the cursor on the screen and optionally prints a message at the new location                                                    |
| <a href="#">SCRPUT</a>    | displays a message in color                                                                                                               |
| <a href="#">SENDMAIL</a>  | sends an email message                                                                                                                    |
| <a href="#">SETLOCAL</a>  | saves the current disk drive, default directory, environment, alias list, and special character settings (see <a href="#">ENDLOCAL</a> ). |
| <a href="#">SHIFT</a>     | changes the numbering of the batch file parameters                                                                                        |
| <a href="#">SMPP</a>      | sends messages using the SMPP protocol                                                                                                    |
| <a href="#">SNPP</a>      | sends a message to an alphanumeric pager                                                                                                  |
| <a href="#">START</a>     | starts another session or window                                                                                                          |
| <a href="#">SWITCH</a>    | selects a group of statements to execute based on the value of a variable                                                                 |
| <a href="#">TCTOOLBAR</a> | changes the TC tool bar buttons                                                                                                           |
| <a href="#">TEXT</a>      | displays a block of text (see <a href="#">ENDTEXT</a> )                                                                                   |
| <a href="#">TIMER</a>     | starts or reads a stopwatch                                                                                                               |
| <a href="#">TITLE</a>     | changes the window title                                                                                                                  |
| <a href="#">VSCRPUT</a>   | displays a vertical message in color                                                                                                      |
| <a href="#">WMIQUERY</a>  | Query the Windows Management Instrumentation interface                                                                                    |

These commands, along with the internal variables and variable functions, make the enhanced batch file language extremely powerful.

#### 4.6.2.8 Interrupting a Batch File

You can usually interrupt a batch file by pressing **Ctrl-C** or **Ctrl-Break**. Whether and when these keystrokes are recognized will depend on whether **TCC** or an application program is running, how the application, if any, was written, whether [BREAK](#) is ON or OFF, and whether the [ON BREAK](#) command is

in use.

If **TCC** detects a **Ctrl-C** or **Ctrl-Break** when ON BREAK is not in use, it displays a prompt, for example:

```
Cancel batch job C:\CHARGE.BTM ? (Y/N/A) :
```

Enter **N** to continue, **Y** to terminate the current batch file and continue with any batch file which called it, or **A** to end all batch file processing regardless of the batch file nesting level. Answering **Y** is similar to the [QUIT](#) command; answering **A** is similar to the [CANCEL](#) command.

#### 4.6.2.9 Detecting TCC and Take Command

From a batch file, you can determine if **TCC** is loaded by doing a numeric comparison:

```
if 01 == 1 echo Take Command is loaded!
```

In **TCC**, this is a numeric comparison and true; in CMD it is a string comparison and false. Once you have established that the batch file is running in **TCC**, you can use internal variables like \_CMDPROC, \_4VER, \_DOS, \_DOSVER, and \_WIN to further determine the operating environment.

You can determine if **TCC** is running in a **Take Command** tab window with the internal variable [\\_TCTAB](#):

```
if %_tctab == 1 echo TCC is running in a Take Command tab window!
```

You can prevent your batch file from running in CMD by giving it the .BTM extension. CMD doesn't recognize .BTM files as batch files.

#### 4.6.2.10 Using Aliases in Batch Files

One way to simplify batch file programming is to use aliases to hide unnecessary detail inside a batch file. For example, suppose you want a batch file to check for certain errors, and display a message and exit if one is encountered. This example shows one way to do so:

```
setlocal
unalias *
setdos /e%^ /c%=& /p%=$
alias error `echo. & echo ERROR: %$ & goto dispmenu`
alias fatalerror `echo. & echo FATAL ERROR: %$ & quit`
alias in `pushd %1 & %2$ & popd`
if not exist setup.btm fatalerror Missing setup file!
call setup.btm
cls
:dispmenu
text
 1. Word Processing
 2. Solitaire
 3. Internet
 4. Exit
endtext
echo.
inkey Enter your choice: %%userchoice
switch %userchoice
case 1
 input Enter the file name: %%fname
```

```
 if not exist fname error File does not exist
 in d:\letters c:\windows\wordpad.exe
case 2
 in d:\finance c:\windows\sol.exe
case 3
 in d:\comm c:\windows\iexplore.exe
case 4
 goto done
default
 error Invalid choice, try again
endswitch
goto dispmenu
:done
endlocal
```

The first alias, ERROR, simply displays an error message and jumps to the label DISPMENU to redisplay the menu. The %\$ in the second [ECHO](#) command displays all the text passed to ERROR as the content of the message. The similar FATALERROR alias displays the message, then exits the batch file.

The last alias, IN, expects 2 or more command line parameters. It uses the first as a new working directory and changes to that directory with a [PUSHD](#) command. The rest of the command line is interpreted as another command plus possible command line parameters, which the alias executes. This alias is used here to switch to a directory, run an application, and switch back. It could also be used from the command line.

The following 9 lines print a menu on the screen and then get a keystroke from the user and store the keystroke in an environment variable called [userchoice](#). Then the [SWITCH](#) command is used to test the user's keystroke and to decide what action to take.

There's another side to aliases in batch files. If you're going to distribute your batch files to others, you need to remember that they may have aliases defined for the commands you're going to use. For example, if the user has aliased [CD](#) to [CDD](#) and you aren't expecting this, your file may not work as you intended. There are two ways to address this problem.

The simplest method is to use [SETLOCAL](#), [ENDLOCAL](#), and [UNALIAS](#) to clear out aliases before your batch file starts, and [SETDOS](#) to select the special characters you depend on, and restore them at the end, as we did in the previous example. Remember that [SETLOCAL](#) and [ENDLOCAL](#) will save and restore not only the aliases but also the environment, the current drive and directory, and various special characters.

If this method isn't appropriate or necessary for the batch file you're working on, you can also use an asterisk \* before the name of any command. The asterisk means the command that follows it should not be interpreted as an alias. For example the following command redirects a list of file names to the file *FILELIST*:

```
dir /b > filelist
```

However, if the user has redefined DIR with an alias this command may not do what you want. To get around this just use:

```
*dir /b > filelist
```

The same can be done for any command in your batch file. If you use the asterisk, it will disable alias processing, and the rest of the command will be processed normally as an internal command, external command, or batch file. Using an asterisk before a command will work whether or not there is actually an alias defined with the same name as the command. If there is no alias with that name, the asterisk will be ignored and the command will be processed as if the asterisk wasn't there.

You can use the pseudovariables [%\\_](#) and [%+](#) to represent the [command escape](#) and [command separator](#) characters, respectively. There is no pseudovariable for the [parameter character](#).

#### 4.6.2.11 Debugging Batch Files

**Take Command** includes a built-in full-featured batch file debugger invoked with the [BDEBUGGER](#) command. The debugger gives you a detailed, step-by-step view of batch file execution, and will help solve particularly difficult batch file problems.

#### 4.6.2.12 String Processing

As you gain experience with batch files, you're likely to find that you need to manipulate text strings. You may need to prompt a user for a name or password, process a list of files, or find a name in a phone list. All of these are examples of string processing -- the manipulation of readable text.

**TCC** includes several features that make string processing easier. For example, you can use the [INPUT](#), [MSGBOX](#), and [QUERYBOX](#) commands for user input; the [ECHO](#) and [ECHOERR](#), [ECHOS](#) and [ECHOSERR](#), [SCREEN](#), [SCRPUT](#), and [VSCRPUT](#) commands for output; and the [FOR](#) command or the [@FILEREAD](#) function to scan through the lines of a file. In addition, [variable functions](#) offer a wide range of [strings and character handling](#) capabilities.

For example, suppose you need a batch file that will prompt a user for a name, break the name into a first name and a last name, and then run a hypothetical LOGIN program. LOGIN expects the syntax **/F:first /L:last** with both the first and last names in upper case and neither name longer than 8 characters. Here is one way to write such a batch file:

```
@echo off
setlocal
unalias *
input Enter your name (no initials): %%name

set first=%@word[0,%name]
setflen=%@len[%first]
set last=%@word[1,%name]
set llen=%@len[%last]

iff %flen gt 8 .or. %llen gt 8 then
 echo First or last name too long
 quit
endiff

login /F:@upper[%first] /L:@upper[%last]
endlocal
```

The [SETLOCAL](#) command at the beginning of this batch file saves the environment and aliases. Then the [UNALIAS \\*](#) command removes any existing aliases so they won't interfere with the behavior of the commands in the remainder of the batch file. The first block of lines ends with a [INPUT](#) command which asks the user to enter a name. The user's input is stored in the environment variable NAME.

The second block of lines extracts the user's first and last names from the NAME variable and calculates the length of each. It stores the first and last name, along with the length of each, in additional environment variables. Note that the [@WORD](#) function numbers the first word as 0, not as 1.

The [IFF](#) command in the third block of lines tests the length of both the first and last names. If either is longer than 8 characters, the batch file displays an error message and ends. (QUERYBOX can limit the length of input text more simply with its [/L](#) switch. We used a slightly more cumbersome method above in order to demonstrate the use of string functions in batch files.)

Finally, in the last block, the batch file executes the LOGIN program with the appropriate parameters, then uses the [ENDLOCAL](#) command to restore the original environment and alias list. At the same time, ENDLOCAL discards the temporary variables that the batch file used (NAME, FIRST, FLEN, etc.).

When you're processing strings, you also need to avoid some common traps. The biggest one is handling special characters.

Suppose you have a batch file with these two commands, which simply accept a string and display it:

```
input Enter a string: %%str
echo %str
```

Those lines look safe, but what happens if the user enters the string "some > none" (without the quotes). After the string is placed in the variable STR, the second line becomes

```
echo some > none
```

The ">" is a [redirection](#) symbol, so the line echoes the string "some" and redirects it to a file called NONE -- probably not what you expected. You could try using [double quotes](#) to avoid this kind of problem, but that won't quite work. If you use back-quotes (ECHO `%STR`), the command will echo the four-character string %STR. Environment variable names are not expanded when they are inside back-quotes.

If you use double quotes (ECHO "%STR"), the string entered by the user will be displayed properly, and so will the double quotes. With double quotes, the output would look like this:

```
"some > none"
```

As you can imagine, this kind of problem becomes much more difficult if you try to process text from a file. Special characters in the text can cause all kinds of confusion in your batch files. Text containing back-quotes, double quotes, or redirection symbols can be virtually impossible to handle correctly.

One way to overcome these potential problems is to use the [SETDOS /X](#) command to temporarily disable redirection symbols and other special characters. The two-line batch file above would be a lot more likely to produce the expected results if it were rewritten this way:

```
setdos /x-15678
input Enter a string: %%str
echo %str
setdos /x0
```

The first line turns off alias processing and disables several special symbols, including the command separator and all redirection symbols. Once the string has been processed, the last line re-enables the

features that were turned off in the first line.

If you need advanced string processing capabilities beyond those provided by **TCC**, you may want to consider using the [Perl](#), [Python](#), [REXX](#), or [Ruby](#) languages. Our products can execute Perl, Python, REXX, and Ruby programs internally, and also support evaluating individual Perl, Python, REXX, and Ruby expressions internally.

#### 4.6.2.13 Batch File Compression

You can compress your *.BTM* files with [BATCOMP](#). That command reduces the size of large batch files by at least a half and makes them unreadable with the [LIST](#) command and similar utilities. Compressed batch files run at approximately the same speed as uncompressed *.BTM* files.

You may want to consider compressing batch files if you need to distribute them to others and keep your original code secret or prevent your users from altering them. You may also want to consider compressing batch files to save some disk space on the systems where compressed files are used.

The full syntax for the batch compression command is

```
BATCOMP [/Q] [/O] InputFile OutputFile
```

You must specify the full name of the input file and output files, including their extensions, on the BATCOMP command line. For example, to compress *MYBATCH.CMD* and save the result as *MYBATCH.BTM*, you use this command:

```
batcomp mybatch.cmd mybatch.btm
```

If the output file (*MYBATCH.BTM* in the examples above) already exists, BATCOMP will prompt you before overwriting the file. You can disable the prompt by including */O* on the BATCOMP command line immediately before the input file name. Even if you use the */O* option, BATCOMP will not compress a file into itself.

The */Q* ("quiet") option suppresses informational messages from BATCOMP.

JP Software does not provide a utility to decompress batch files. If you use BATCOMP, make sure that you also keep a copy of the original batch file for future inspection or modification.

If you plan to distribute batch files to users of different platforms, see [Special Character Compatibility](#) for important information on the command separator, escape character, and parameter character used in each product.

#### 4.6.2.14 Lua support

Lua is a powerful, fast, lightweight, embeddable scripting language. **TCC** includes internal support for Lua, both for executing Lua scripts and for executing individual Lua expressions. The version supplied with **TCC** is Lua 5.2. For more information on Lua, go to <http://www.lua.org>.

You must enable Lua support in the [OPTION](#) / Startup page. If it is enabled, **TCC** will automatically load LUA.DLL on your system. **TCC** checks to see if you are running a *.LUA* file. If so, **TCC** passes the file to the Lua interpreter for processing.

See also: the [@LUA](#) function.

#### 4.6.2.15 Perl support

Perl is a powerful file and text processing language available on many platforms. Perl is a useful extension to the **TCC** batch language, especially if you need advanced string processing capabilities.

The Perl language is not built into **TCC**, and must be obtained separately. The version supported by **TCC** is PerlScript (the WSH COM interface), which is included in Active State Perl (free from [www.activestate.com](http://www.activestate.com)).

You must enable Perl support in the [OPTION](#) / Startup page. If it is enabled, **TCC** will automatically load Perl on your system. If a suitable library is found, **TCC** checks to see if you are running a **.PL** file. If so, **TCC** passes the file to your Perl interpreter for processing.

See also: the [@PERL](#), [@PYTHON](#), [@REXX](#) and [@RUBY](#) functions.

#### 4.6.2.16 Python support

Python is a powerful file and text processing language available on many platforms. Python is an ideal extension to the **TCC** batch language, especially if you need advanced string processing capabilities.

The Python language is not built into **TCC**, and must be obtained separately. The version supported by **TCC** is ActivePython (free from [www.activestate.com](http://www.activestate.com)). TCC supports version 3.1, 2.6, and 2.5. (TCC will search for the Python dll's in that order.)

You must enable Python support in the [OPTION](#) / Startup page. If it is enabled, **TCC** will automatically load a Python interpreter when it starts. If a suitable library is found, **TCC** checks to see if you are running a **.PY** file. If so, **TCC** passes the file to your Python interpreter for processing.

See also: the [@PYTHON](#), [@PERL](#), [@REXX](#) and [@RUBY](#) functions.

#### 4.6.2.17 REXX Support

REXX is a powerful file and text processing language developed by IBM, and available on many platforms. REXX is an ideal extension to the **TCC** batch language, especially if you need advanced string processing capabilities.

The REXX language is not built into **TCC**, and must be obtained separately by downloading the free ooREXX (Open Object REXX) from <http://www.oorexx.org/>.

You must enable REXX support in the [OPTION](#) / Startup page. If it is enabled, when **TCC** loads it asks Windows to locate specific REXX libraries associated with Open Object REXX. If ooREXX is found, **TCC** checks to see if you are running a **.REX** or **.REXX** file, or if the first two characters on the first line of a **.CMD** file are **[/\*]**, the beginning of a REXX comment. If either of these tests succeeds, **TCC** passes the file to your REXX interpreter for processing.

When you send a command from a REXX program back to **TCC** to be executed (for example, if you execute a DIR command within a REXX script), the REXX software must use the correct address for **TCC**. **TCC** uses the address **CMD** for compatibility with scripts written for CMD.

For details on communication between REXX and **TCC**, or for more information on any aspect of REXX, see your ooREXX documentation.

See also: the [@REXX](#), [@PERL](#), [@PYTHON](#), and [@RUBY](#) functions.

#### 4.6.2.18 Ruby support

Ruby is a powerful object-oriented file and text processing language available on many platforms. Ruby is an ideal extension to the **TCC** batch language, especially if you need advanced string processing capabilities.

The Ruby language is not built into **TCC**, and must be obtained separately. The versions supported by **TCC** are Ruby 1.8 and 1.9 (free from [www.ruby-lang.org](http://www.ruby-lang.org) ).

You must enable Ruby support in the **OPTION** / Startup page. If it is enabled, **TCC** will automatically load the Ruby library at startup. If a suitable library is found, **TCC** checks to see if you are running a **.rb** file. If so, **TCC** passes the file to your Ruby interpreter for processing.

See also: the [@RUBY](#), [@PYTHON](#), [@REXX](#) and [@PERL](#) functions.

#### 4.6.2.19 Tcl/tk Support

The Tcl/tk language is not built into **TCC**, and must be obtained separately. The version supported by **TCC** is ActiveTcl 8.5.7 (free from [www.activestate.com](http://www.activestate.com)).

You must enable Tcl support in the **OPTION** / Startup page. If it is enabled, **TCC** will automatically load a Tcl interpreter when it starts. If a suitable library is found, **TCC** checks to see if you are running a **.TCL** file. If so, **TCC** passes the file to your Tcl interpreter for processing.

It's not possible for **TCC** to determine in advance whether you're running a Tcl or a Tk script. After executing the script, **TCC** checks if a Tk window is running. If so, it enters a Tk event loop and waits for the window to be closed. If not, **TCC** assumes it was a Tcl script and **TCC** returns immediately.

Because of the way the Tk interpreter works, it is not possible for **TCC** to maintain a persistent interpreter after executing a Tk script. **TCC** will close the current Tcl/tk interpreter and create a new one the next time a Tcl / tk script is executed.

See also [@TCL](#) and [@TK](#).

#### 4.6.2.20 EXTPROC / SHEBANG Support

**TCC** offers an external processor option for batch files that lets you define an external program to process a particular **.CMD** file. To identify a **.CMD** file to be used with an external processor, place the string **EXTPROC** as the first word on the first line of the file, followed by the name of the external program that should be called. **TCC** will start the program and pass it the name of the **.CMD** file and any command line parameters that were entered.

For example, suppose **GETDATA.CMD** contains the following lines:

```
EXTPROC D:\DATAACQ\DATALOAD.EXE
OPEN PORT1
READ 4000
DISKWRITE D:\DATAACQ\PORT1\RAW
```

Then if you entered the command:

```
[d:\dataacq] getdata /p17
```

**TCC** would read the **GETDATA.CMD** file, determine that it began with an EXTPROC command, read the name of the processor program, and then execute the command:

```
D:\DATAACQ\DATALOAD.EXE D:\DATAACQ\GETDATA.CMD /p17
```

The hypothetical *DATALOAD.EXE* program would then be responsible for reopening the *GETDATA.CMD* file, ignoring the EXTPROC line at the start, and interpreting the other instructions in the file. It would also have to respond appropriately to the command line parameter entered (/p17).

Do not try to use **TCC** as the external processor named on the EXTPROC line in the *.CMD* file. It will interpret the EXTPROC line as a command to reopen itself. The result will be an infinite loop that will continue until the computer runs out of resources and locks up.

**TCC** also provides **SHEBANG** support. It works identically to **EXTPROC**, but the first line begins with a **#!**.

Note that **EXTPROC** and **SHEBANG** only work with files with a *.CMD* extension, not *.BTM* or *.BAT*.

## 4.7 File Selection

Most internal commands (like [COPY](#), [DIR](#), etc.) work on a file or a group of files. You can use several shorthand forms for naming or selecting files and the applications associated with them, or for accessing files on remote systems.

Most of the features explained in this section apply to **TCC** commands only, and generally cannot be used to pass file names to external programs (unless those programs were specifically written to support these features).

The features discussed in this section are:

- [Wildcards](#)
- [Executable Extensions](#)
- [Using Internet URLs](#)
- [Using FTP and HTTP Servers](#)
- [OpenAFS](#)
- [Ranges](#)
- [Attribute Switches](#)
- [Multiple Filenames](#)
- [Include Lists](#)
- [Delayed Variable Expansion](#)
- [Extended Parent Directory Names](#)
- [LFN File Searches](#)
- [@File Lists](#)
- [Command Switches for File Selection](#)

### 4.7.1 Wildcards and Regular Expressions

Wildcards let you specify a file or group of files by typing a partial filename. The appropriate directory is scanned to find all of the files that match the partial name. You can also specify files with [regular expressions](#).

Wildcards are usually used to specify which files should be processed by a command. If you need to specify which files should not be processed, see [File Exclusion Ranges](#) (for internal commands), or [EXCEPT](#) (for external commands).

Most internal commands accept filenames with wildcards anywhere that a full filename can be used.

There are two wildcard characters, the [asterisk](#) \* and the [question mark](#) ?. Additionally, you can specify a [set of characters](#). Note the issues about [matching short file names](#).

**WARNING:** When you use a wildcard search for files to process in a command like [FOR](#) or [DO](#), and you create new filenames (whether by renaming existing files or by creating new files), the new filenames may match your selection wildcard, and cause you to process them again.

**TCC** (but not **TCC/LE**) also supports wildcards in the directory names (but not in the drive name). You can control the subdirectory recursion by specifying \* or \*\* in the path. A \* will match a single subdirectory level; a \*\* will match any all subdirectory levels for that pathname. Directory wildcards also support regular expressions. Directory wildcards cannot be used with the /O:.... option (which sorts entries before executing the command). And think very carefully before using directory wildcards with a /S (recurse subdirectories) option, as this will almost certainly return unexpected results! There are a few commands which do not support directory wildcards, as they would be meaningless or destructive (for example, TREE, @FILEOPEN, @FILEDATE, etc.).

For example:

|                             |                                                                                                                                               |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| del c:\test\test2\*\foobar  | Delete the file <b>foobar</b> in any subdirectory of <b>c:\test\test2</b> (but not in any of their subdirectories).                           |
| del c:\test\**\*foo*\foobar | Delete the file <b>foobar</b> in any subdirectory under <b>c:\test</b> (and all of their subdirectories) that has "foo" anywhere in the name. |
| del c:\test\t*2\foobar      | Delete the file <b>foobar</b> in any subdirectory of <b>c:\test</b> that begins with a <b>t</b> and ends with a <b>2</b> .                    |

### Asterisk \* wildcard

An asterisk \* in a file specification means "a set of any characters or no character in this position". For example, this command will display a list of all files (including directories, but excluding those files and directories with at least one of the attributes *hidden* and *system*) in the current directory:

```
dir *
```

If you want to see all of the files with a **.TXT** extension:

```
dir *.txt
```

If you know that the file you are looking for has a base name that begins with **ST** and an extension that begins with **.D**, you can find it this way. Filenames such as **STATE.DAT**, **STEVEN.DOC**, and **ST.D** will all be displayed:

```
dir st*.d*
```

**TCC** also lets you also use the asterisk to match filenames with specific letters somewhere inside the name. The following example will display any file with a **.TXT** extension that has the letters **AM** together anywhere inside its base name. It will, for example, display **AMPLE.TXT**, **STAMP.TXT**, **CLAM.TXT**, and **AM.TXT**, but it will ignore **CLAIM.TXT**:

```
dir *am*.txt
```

### Question mark ? wildcard

A question mark ? matches any single filename character. You can put the question mark anywhere in a filename and use as many question marks as you need. The following example will display files with names like *LETTER.DOC*, *LATTER.DAT*, and *LITTER.DU*:

```
dir l?tter.d??
```

The use of an asterisk wildcard before other characters, and of the character ranges discussed below, are enhancements to the standard Microsoft wildcard syntax, and are not likely to work properly with software other than **TCC**.

"Extra" question marks in your wildcard specification are ignored if the file name is shorter than the wildcard specification. For example, if you have files called *LETTER.DOC*, *LETTER1.DOC*, and *LETTERA.DOC*, this command will display all three names:

```
dir letter?.doc
```

The file *LETTER.DOC* is included in the display because the "extra" question mark at the end of **LETTER?** is ignored when matching the shorter name *LETTER*.

### Specific character set

In some cases, the ? wildcard may be too general. **TCC** also allows you to specify the exact set of what characters you want to accept (or exclude) in a particular position in the filename by using square brackets [ ]. Inside the brackets, you can put the individual acceptable characters or ranges of characters. For example, if you wanted to match *LETTER0.DOC* through *LETTER9.DOC*, you could use this command:

```
dir letter[0-9].doc
```

You could find all files that have a vowel as the second letter in their name this way. This example also demonstrates how to mix the wildcard characters:

```
dir ?[aeiouy]*
```

You can exclude a group of characters or a range of characters by using an exclamation mark (!) as the first character inside the brackets. This example displays all filenames that are at least 2 characters long except those which have a vowel as the second letter in their names:

```
dir ?[!aeiouy]*
```

The next example, which selects files such as *AIP*, *BIP*, and *TIP* but not *NIP*, demonstrates how you can use multiple ranges inside the brackets. It will accept a file that begins with an *A*, *B*, *C*, *D*, *T*, *U*, or *V*:

```
dir [a-dt-v]ip
```

You may use a question mark character inside the brackets, but its meaning is slightly different than a normal (unbracketed) question mark wildcard. A normal question mark wildcard matches any character, but will be ignored when matching a name shorter than the wildcard specification, as described above. A question mark inside brackets will match any character, but will **not** be discarded when matching shorter filenames. For example:

```
dir letter[?].doc
```

will display *LETTER1.DOC* and *LETTERA.DOC*, but not *LETTER.DOC*.

You can repeat any of the wildcard characters in any combination you desire within a single file name. For example, the following command lists all files which have an **A**, **B**, or **C** as the third character, followed by zero or more additional characters, followed by a **D**, **E**, or **F**, followed optionally by some additional characters, and with an extension beginning with **P** or **Q**. You probably won't need to do anything this complex, but we've included it to show you the flexibility of extended wildcards:

```
dir ??[abc]*[def]*.[pq]*
```

You can also use the square bracket wildcard syntax to work around a conflict between long filenames containing semicolons [;], and the use of a semicolon to indicate an [include list](#). For example, if you have a file on an LFN drive named C:\DATA\LETTER1;V2 and you enter this command:

```
del \data\letter1;v2
```

you will not get the results you expect. Instead of deleting the named file, **TCC** will attempt to delete *LETTER1* and then *V2*, because the semicolon indicates an [include list](#). However if you use square brackets around the semicolon it will be interpreted as a filename character, and not as an include list separator. For example, this command would delete the file named above:

```
del \data\letter1[;]v2
```

### Matching short file names

If the [Search for SFNs](#) configuration option is set, wildcard searches accept a match on either the LFN or the SFN to match the behavior of CMD. This may cause some files to be found because of SFN match only. In most situations this is not actually desirable, and can be avoided by disabling the option (the default).

**Note:** The wildcard expansion process will attempt to allow both CMD-style "extension" matching (only one extension, at the end of the word) and the advanced **TCC** filename matching (allowing things like \*.\*.abc) when an asterisk is encountered in the destination of a [COPY](#), [MOVE](#) or [REN/RENAME](#) command.

### Regular Expressions

You can also use [regular expressions](#) for file name tests. (The type of regular expressions to use is specified by the [Regular Expressions Syntax](#) option.)

The syntax is:

```
::regex
```

For example:

```
dir ::ca[td]
```

Note that using regular expressions will slow your directory searches -- since Windows doesn't support them, the parser has to convert the filename to \*, retrieve all filenames, and then match them to the expression.

If you have any special characters (whitespace, redirection characters, escape characters, etc.) in your regular expression, you will need to enclose it in double quotes. For example:

```
dir ":::^\\w{1,8}\\.btm$"
```

For more information on the syntax, see [Regular Expression Syntax](#).

#### 4.7.2 Executable Extensions

Normally, when you type a filename (as opposed to an alias or internal command name) as the first word on the command line, **TCC** looks for a file with that name to execute.

The file's extension may be *.EXE* to indicate that it contains a program; *.LNK* to indicate that it contains information on how to execute a program under Windows; or *.BTM*, *.BAT*, or *.CMD* to indicate a [batch file](#).

You can add to the default list of extensions, and have **TCC** take the action you want with files that are not executable programs or batch files. The action taken is always based on the file's extension. For example, you could start your text editor whenever you type the name of a *.DOC* file, or start your database manager whenever you type the name of a *.DAT* file.

Windows also includes the ability to associate file extensions with specific applications. See [Windows File Associations](#) for details on this feature and its relationship to executable extensions. See also: [Executable Files and File Searches](#).

You use environment variables to define the internal command, external program, batch file, or alias to run for each defined file extension. To create an executable extension for use only in **TCC**, use the [SET](#) command to create a new environment variable. An environment variable is recognized as an executable extension if its name begins with a period.

The syntax for creating an executable extension is:

```
set .ext[;.ext[;...]]=command [options]
```

where *.EXT* is the executable file extension; **command** is the name of the internal command, alias, external program, or batch file to run; and **[options]** are any command line startup options you want to specify for the program, batch file, or alias. You can specify multiple extensions for a single command by separating them with semicolons.

For example, if you want to run a word processor called *EDITOR* whenever you type the name of a file that has an extension of *.EDT*, you could use this command:

```
set .edt=c:\\edit\\editor.exe
```

If the command specified in an executable extension is a batch file or external program, **TCC** will search the [PATH](#) for it if necessary. However, you can make sure that the correct program or batch file is used, and speed up the executable extension, by specifying the full name including drive, path, filename, and extension. You can utilize other environment variables in the specification.

Once an executable extension is defined, any time you name a file with that extension as a command, it is equivalent to having typed the value of the extension variable, followed by the name of the file.

The next example defines *WORDPAD.EXE* (a Windows editor) as the processor for *.TXT* files:

```
set .txt="c:\program files\accessories\wordpad.exe"
```

Now, if you have a file called *HELLO.TXT* and enter the command

```
hello
```

**TCC** will execute the command:

```
"c:\program files\accessories\wordpad.exe" c:\source\hello.txt
```

Notice that the full pathname of *HELLO.TXT* is automatically included. If you enter parameters on the command line, they are appended to the end of the command. For example, if you changed the above entry to:

```
[c:\source] hello -w
```

**TCC** would execute the command:

```
"c:\program files\accessories\wordpad.exe" c:\source\hello.txt -w
```

In order for executable extensions to work, the command, program, batch file, or alias must be able to interpret the command line properly. For example, if a program you want to run doesn't accept a file name on its command line as shown in these examples, then executable extensions won't work with that program.

Executable extensions may include [wildcards](#), so you could, for example, run your text editor for any file with an extension beginning with *T* by defining an executable extension called *.T\**. Extended wildcards (e.g., **DO[CT]** for *.DOC* and *.DOT* files) may also be used.

To remove an executable extension, use [UNSET](#) to remove the corresponding variable.

#### 4.7.3 Using Internet URLs

If you type an Internet URL (Uniform Resource Locator) which begins with **http:** or **https:** at the prompt, **TCC** will pass the URL to Windows. Normally Windows will start your web browser, and request that the browser retrieve the page pointed to by the URL. This feature will only work if Windows can find the proper association between the **http:** or **https:** prefix and the browser software. While this association is standard for most browser installations, it may not be present on all systems.

The ability to "start" URLs in this way is restricted to those beginning with **http:** or **https:**. Other standard prefixes such as **ftp:**, **mail:**, and **news:** cannot be started directly from the prompt; you must enter these URLs directly into your browser.

HTTP and HTTPS addresses in **Take Command** and **TCC** will have any embedded spaces converted to "%20" before sending the URL to the server.

See [Waiting for Applications to Finish](#) for information on problems with waiting for the browser to finish after starting a URL.

#### 4.7.4 Using FTP and HTTP Servers

**TCC** allows direct access to remote servers from internal commands such as [COPY](#), [DEL](#), [DIR](#), [MOVE](#), [MD](#), [RD](#), [REN](#), and [SELECT](#) via several protocols:

- [FTP](#) (basic FTP)
- [TFTP](#) (Trivial FTP)
- [FTPS](#) (SSL FTP)
- [SFTP](#) (SSH FTP)
- [HTTP](#) (basic Web access)
- [HTTPS](#) (SSL HTTP)

**Note:** Not all protocols are supported in every internal command. For example, DIR will not work with HTTP or HTTPS (because of limitations in the HTTP / HTTPS protocol).

- **FTP support:**

The basic filename syntax for anonymous connections is:

```
ftp://ftp.abc.com/...
```

For example, to get a directory of the Microsoft FTP site, you could use this command:

```
dir ftp://ftp.microsoft.com/*
```

If you don't specify a username and password, **TCC** will look for your FTP user names and passwords in the file *FTP.CFG* (which defaults to the **Take Command** directory). You can specify another directory with the [FTP.CFG](#) configuration option. You must add entries to the *FTP.CFG* file manually. The format for each line is:

```
url [(alias)] username password [directory template]
```

For example:

```
jisoft.com fred secret
microsoft.com anyone mypassword
```

You can have multiple users for a single FTP site (for example, an admin user and a normal user). You need to add an alias (enclosed in parentheses) following the name of the ftp site. For example:

```
jisoft.com (jadmin) Bob AdminPassword
jisoft.com (jpublic) anonymous Bob@ftp.jisoft.com
```

You can then access the server as `ftp://jadmin` or `ftp://jpublic`.

We recommend you encrypt this file if you're using NTFS. If *FTP.CFG* doesn't exist the first time **TCC** looks for it, it will be created as an encrypted file (NTFS only). **Note:** If you are using FAT / VFAT, the file will not be encrypted and your user names and passwords will be unprotected in plain text.

You can also specify an explicit username and password on the command line:

```
ftp://[username:password@]ftp.abc.com/...
```

If you specify a password of \*, you will be prompted to enter the password (which will appear on the screen as asterisks). Depending on the type of operation you're doing, you may need to enter the password multiple times as **TCC** repeatedly connects and disconnects from the server. To avoid this, use [IFTP](#) (which will also be much faster).

If you have FTP permission on server **ftp.abc.com** and a subdirectory of the root directory on that server is called *mydir*, you can display the files with this command (enter this on one line):

```
dir ftp://username:password@ftp.abc.com/mydir/*
```

You can also use the internal **IFTP** command to start an FTP session with a server and then use a simplified syntax to manipulate files on the server.

**TCC** also supports symbolic hostnames (defined in \windows\system32\drivers\etc\hosts).

**TCC** normally connects to the FTP server on the default FTP port 21. If the FTP server you are connecting to uses a non-standard port, enter the port number (with a preceding colon) just after the server name, for example:

```
dir ftp://username:password@ftp.abc.com:8765/mydir/*
```

To log on to a server which supports "anonymous" logins, enter the required user name (usually "anonymous") and password (usually your email address) using the syntax shown above, for example:

```
dir ftp://anonymous:email@domain.com@ftp.microsoft.com/
```

**TCC** will distinguish between the @ in the email address and the @ before the server name in order to separate the parts of the URL properly.

If you use a partial file or path reference, such as

```
dir ftp:myfile.txt
```

**TCC** will attempt to build a fully qualified directory name in which to find the requested file or path, based on what the server reports as the current working directory. If an ftp file or path specification begins with a ~ (tilde, typically indicative of a path relative to the user's home directory), **TCC** will instead pass the exact string directly to the remote server.

**TCC** uses standard FTP commands to retrieve information about files and directories and manipulate those files and directories on FTP servers, and relies on the server's compliance with Internet FTP standards. If your server is not fully compliant, or does not operate in the manner that **TCC** expects, the results may not be what you intend. For example, if the FTP server you are connecting to is case-sensitive, you may have to use the stored case of file and directory names when you use FTP commands. (If you include wildcards in the filename, **TCC** will match filenames regardless of case.) We urge you to test each server you use with nondestructive commands like DIR before you try to copy or delete files, create or remove directories, etc.

Time-related operations (e.g. switches like **COPY /C** or **/U**) may not always work reliably on FTP and HTTP servers, due to differences in time zone and in the file time representations between your local system and the server. Be sure to experiment with the particular server in question before depending on commands which compare file times to yield the results you want.

**Note:** If you use a partial reference such as **ftp:mydir** outside the scope of an **IFTP** command, **TCC** will attempt to re-establish the last connection, if any. That new connection may or may not be logged to the last used directory on that sever. We recommend you always use a full reference (including server name) unless you are specifically taking advantage of an active **IFTP** connection. You can determine if there is an active **IFTP** connection with the **\_ftp** and **\_https** variables.

Before you can use the built-in FTP support or the [IFTP](#) command, you must establish the necessary connection to the Internet. For example, if you use Windows Dial-Up Networking to connect to the Internet, you must start your dial up connection first. If you connect through a proxy server, you must set the [Proxy](#) configuration options.

#### Non-standard FTP servers:

**TCC** supports directory formats for the following:

EPLF  
WFTP  
VMS (single-line filenames only)  
NetPresenz (Macintosh)  
Netware  
All known UNIX and Linux formats  
Windows FTP Server

If you have a non-standard FTP server that creates an unusual directory format, you can create an entry in your *FTP.CFG* file to allow **TCC** to parse the FTP server output. The format is described in the *FTP.CFG* following the host name, username, and password. The format characters are:

- I"text"** Do a wildcard comparison of "text" and the directory line; if it matches, discard the entire line. (This is to allow you to skip header & footer lines that would otherwise return garbage.)
- "text"** Compare (and skip) a literal string (does NOT support wildcard searches)
- <space>** Skip whitespace (spaces, tabs)
- !** Skip non-whitespace
- Ignore a single character
- F** Filename. If the F is followed by a . (i.e., "F."), the extension is the next non-whitespace string. The extension will be appended (preceded by a '.') to the filename.
- S** Subdirectory flag. If the S is followed by a =, the next character is the character in a "raw" directory listing that denotes a directory. If you don't specify a =, 'D' is assumed.
- T** Month as a string (i.e., "Jan", "Feb", etc.)
- U** Linux-style year (2004) or time (18:30) in the same field.
- Y** Year
- M** Month
- D** Day
- h** Hour
- m** Minute

**H** a or p (for am/pm)

**Z** File size. If the Z is followed by a =, the number following that is the block size.

(Note that upper/lower case is significant for the format characters.)

For example, the FTP.CFG entry for JPSOFT.COM can be described as:

```
jpsoft.com anonymous JPUser@ S !!!Z T D U F
```

- **TFTP ("trivial FTP") support:**

See the [FTP](#) section above for general notes and requirements.

**TFTP** is only available with [COPY](#) (and with [MOVE](#) when the source is a local file). The syntax is:

```
tftp://server[:port]/filename
```

For example:

```
copy update tftp://190.189.188.0/update
```

- **HTTP ("basic Web") support:**

See the [FTP](#) section above for general notes and requirements.

The **HTTP** syntax is:

```
http://[user:password@]server[:port]/filename
```

For example:

```
copy http://jpsoft.com/
```

- **FTPS ("SSL FTP") support:**

See the [FTP](#) section above for general notes and requirements.

The **FTPS** syntax is:

```
ftps://[user:password@]server[:port]/filename
```

For example:

```
copy ftps://bob:pass@ftp.myserver.com/tcmd/tcmd.exe
```

- **SFTP ("SSH FTP") support:**

See the [FTP](#) section above for general notes and requirements.

The **SFTP** syntax is:

```
sftp://[user:password@]server[:port]/filename
```

For example:

```
copy sftp://bob:pass@ftp.myserver.com/tcmd/tcmd.exe
```

- **HTTPS ("SSL HTTP") support:**

See the [FTP](#) section above for general syntax and requirements.

The **HTTPS** syntax is:

```
https://[user:password@]server/filename
```

For example:

```
copy https://jpssoft.com/downloads/v16/tcmd.exe
```

## 4.7.5 OpenAFS

**TCC** has built-in support for OpenAFS. The parser will recognize Linux-style AFS names (i.e., `/afs/athena/user`) and convert them to Windows-compatible names (i.e., `\\\afs\athena\user`). (It will also check for custom AFS mount points, and use that name instead of `afs`.)

See <http://www.openafs.org> for more information on OpenAFS.

## 4.7.6 Ranges

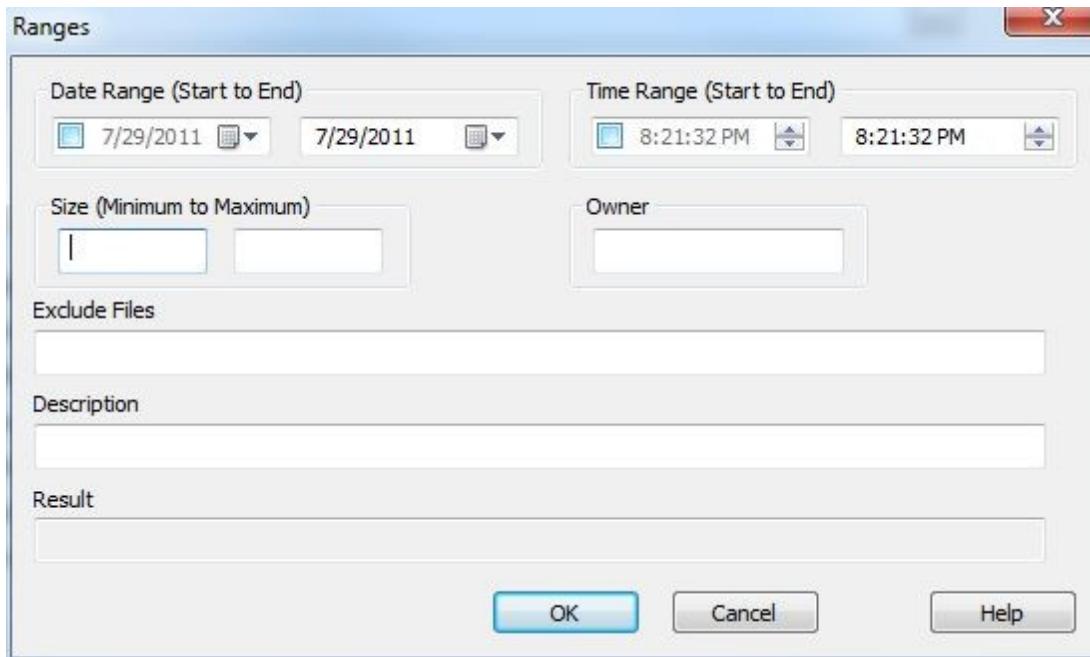
Most internal commands which accept wild cards also allow size, date, time, exclusion, and description ranges to further define the files that you wish to work with. **TCC** will examine each file's properties to determine whether or not the file meets the range criteria that you have specified.

A size, date, time, or exclusion range specification begins with the switch character `/`, followed by a left square bracket `[` and a character that specifies the range type: `s` for size range, `d` for date range, `t` for time range, or `!` for exclusion range. The `s`, `d`, or `t` is followed by a start value, and an optional comma and end value. The range ends with a right square bracket `]`. For example, to select files between 100 and 200 bytes long you could use the range `/[s100,200]`.

A description range begins with `/I`. See [Description Ranges](#) for the full syntax.

If you use the syntax `/[=]`, TCC will display a dialog that allows you to select the ranges you want. (Not included in **TCC/LE**.) For example:

```
copy /[=] file1 file2
```



## General Rules

You can reverse the range test by preceding the range argument with the ! character. For example, to select files that are less than 100 bytes or more than 1000 bytes:

```
/![s100,1000]
```

If you combine different types of ranges, a file must satisfy all range specifications to be included. For example,

```
/ [d2010-2-8,2011-2-9] / [s1024,2048]
```

means files last modified between February 8, 2010 and February 9, 2011, which are also between 1,024 and 2,048 bytes long.

You may not repeat the same range type in a command.

When you use range specifications in a command, they should immediately follow the command name, so that any additional switches for the command are after any range(s) used. If the range is placed later in the command it may be ignored, or cause an error. Unlike some command switches which apply to only part of the command line, the range usually applies to all file names specified for the command. Any exceptions are noted in the descriptions of individual commands.

For example, to get a directory of all the \*.C files dated October 1, 2010, you could use this command:

```
dir / [d2010-10-1,+0] *.c
```

To delete all of the 0-byte files on your disk, you could use this command:

```
del / [s0,0] * /s
```

And to copy all of the non-zero byte files that you changed yesterday or today to your floppy disk, you can use this command:

```
copy /[d-1] /[s1] * a:
```

It can be tedious to type all of the elements of a range, especially when it involves multiple dates and times. In this case you may find it easier to use aliases for common operations. For example, if you often wish to select from .DAT files modified over the last three days and copy the selected files to another drive, you might define an alias like this:

```
alias workback='select /[d-2] copy (*.dat) e:\datfiles\'
```

For more complex requirements, you may want to use internal variables (e.g. [\\_DATE](#) or [\\_TIME](#)) and built-in variable functions (e.g. [@DATE](#), [@TIME](#), [@MAKEDATE](#), [@MAKETIME](#), [@FILEDATE](#), [@FILETIME](#), or [@EVAL](#)). These variables and functions allow you to perform arithmetic and date / time calculations. You may also define your own variable functions, to perform more complex manipulations repetitively.

See the individual types for details on specifying ranges:

- [Size Ranges](#)
- [Date Ranges](#)
- [Time Ranges](#)
- [Exclusion Ranges](#)
- [Description Ranges](#)

Ranges can be used with many commands, including [ATTRIB](#), [COPY](#), [DEL](#), [DESCRIBE](#), [DIR](#), [DO](#), [EXCEPT](#), [FFIND](#), [FOR](#), [HEAD](#), [LIST](#), [MOVE](#), [PDIR](#), [RD](#), [REN](#), [SELECT](#), [TAIL](#), and [TYPE](#)

Ranges cannot be used with filename completion or in filename parameters for variable functions, except as described under the individual functions.

Do not use ranges with [@file](#) lists. See [@file lists](#) for details.

### Date, Time, and Size Ranges

All ranges are inclusive. For example, a size range which selects files from 10,000 to 20,000 bytes long will match files that are exactly 10,000 bytes or 20,000 bytes long, as well as all sizes in between; a date range that selects files last modified between 2010-10-27 and 2010-10-30 will include files modified on each of those dates, and on the two days in between.

If you reverse range start and end values **TCC** will recognize the reversal, and will use the second (lower) value as the start point of the range and the first (higher) value as its end point. For example, to select files between 100 and 200 bytes long could also be entered as [/\[s200,100\]](#).

#### 4.7.6.1 Size Ranges

Size ranges select files whose size is between the inclusive limits specified. The second parameter of a size range is optional. If you use a single parameter, you will select all files of the specified size or larger. You can also precede the second parameter with a plus sign [+]; when you do, it is added to the first value to determine the largest file size to include in the search.

You can exclude a size range by preceding the range with the ! character.

When you use a size range in a command it should immediately follow the command name. See

[General Rules for Using Ranges](#) for additional details.

Either or both values in a size range can be suffixed with a scale factor from the table below. Lower case letters denote a power of 1,000, upper case letters a power of 1,024 ( $2^{**}10$ ).

| Code     | Scale Factor          |             | Code     | Scale Factor          |            | Unit Name |
|----------|-----------------------|-------------|----------|-----------------------|------------|-----------|
| <b>k</b> | 1,000                 | $10^{**3}$  | <b>K</b> | 1,024                 | $2^{**10}$ | kilobyte  |
| <b>m</b> | 1,000,000             | $10^{**6}$  | <b>M</b> | 1,048,576             | $2^{**20}$ | megabyte  |
| <b>g</b> | 1,000,000,000         | $10^{**9}$  | <b>G</b> | 1,073,741,824         | $2^{**30}$ | gigabyte  |
| <b>t</b> | 1,000,000,000,000     | $10^{**12}$ | <b>T</b> | 1,099,511,627,776     | $2^{**40}$ | terabyte  |
| <b>p</b> | 1,000,000,000,000,000 | $10^{**15}$ | <b>P</b> | 1,125,899,906,842,624 | $2^{**50}$ | petabyte  |

Examples of size ranges:

|                     |                                            |
|---------------------|--------------------------------------------|
| Specification       | Selects Files of Length                    |
| <b>/[s0,0]</b>      | zero (empty)                               |
| <b>/[s1M]</b>       | $2^{**20}$ bytes or larger                 |
| <b>/[s10k,+200]</b> | between 10,000 and 10,200 bytes, inclusive |
| <b>/[s10,153k]</b>  | between 10 and 153,000 bytes, inclusive    |
| <b>/![s1K,5K]</b>   | less than 1K or greater than 5K            |

#### 4.7.6.2 Date Ranges

Date ranges select files dated at any time of day between the inclusive limits specified. For example, **/[d2011-12-1,2011-12-5]** selects files that were last modified on *or* after December 1, 2011, but not modified *after* December 5, 2011.

When you use a date range in a command, only other range specifications may be between the command name and the date range. See [General Rules for Using Ranges](#) for additional details.

You can use hyphens, slashes, or periods to separate the month, day, and year. The year can be entered as a 2-digit or 4-digit value. Two-digit years between 80 and 99 are interpreted as 1980...1999; values between 00 and 79 are interpreted as 2000...2079. For example, **/[d2010-12-31,2011-1-1]** selects files modified between December 31, 2010 and January 1, 2011.

If either parameter begins with a four digit year (which must greater than 1900), it is assumed to be a date in the international format **yyyy-mm-dd**, otherwise it is assumed that the date elements are in the order appropriate for your locale. All non-ISO date examples in the HELP use the USA format: mm-dd-yy, unless otherwise stated explicitly.

The default time for the first date is the beginning of that day, and for the second date it is the end of that day. This is true even if the dates are in descending order, i.e., the first date is later than the second one. You can alter these defaults by including specific start and stop times inside the date range. The time is separated from the date with an at sign @. For example, the range **/[d2010-7-01@8:00a,2010-7-03@6:00p]** selects files that were modified at any time between 8:00:00 am on July 1, 2010 and 6:00:00 pm on July 3, 2010. If you prefer, you can specify the times in 24-hour format (e.g., **@18:00** for the end time in the previous example).

If you omit the second parameter in a date range, **TCC** substitutes the current date and time. For example, **/[d2010-10-1]** selects files dated between October 1, 2010 and the instant of command execution.

Instead of an explicit date, you may use an offset value for either the beginning or ending date, or both. An offset begins with a plus sign [+] or a minus sign [-] followed by an integer. If you use an offset for the second value, it is calculated relative to the first. If you use an offset for the first (or only) value, the current date is used as the basis for calculation. For example:

| <b>Specification</b> | <b>Selects Files</b>                                                   |
|----------------------|------------------------------------------------------------------------|
| /[d2010-1-27,+3]     | modified between January 27, 2010 and January 30, 2010                 |
| /[d2010-1-27,-3]     | modified between January 24, 2010 and January 27, 2010                 |
| /[d-0]               | modified today (from today minus zero days, to today)                  |
| /[d-1]               | modified yesterday or today (from today minus one day, to today)       |
| /[d-1,+0]            | modified yesterday (from today minus one day, to zero days after that) |

As a shorthand way of specifying files modified today, you can also use **/[d]**; this has the same effect as the **/[d-0]** example shown above.

Instead of a date, you can specify a file age for the first and/or second parameter (not supported in **TCC/LE**). See [Time Stamps](#), [@AGEDATE](#) and [@MAKEAGE](#).

To select files last modified **n** days ago or earlier, use **/[d-n,1980-1-1]**. For example, to get a directory of all files last modified 3 days or more before today (*i.e.*, those files not modified within the last 3 days), you could use this command:

```
dir /[d-3,1980-1-1]
```

This reversed date range (with the later date given first) will be handled correctly by **TCC**. It takes advantage of the facts that an offset in the start date is relative to today, and that the base or "zero" point for PC file dates is January 1, 1980 for FAT / VFAT, or January 1, 1601 for NTFS.

You cannot use offsets in the time portion of a date range (the part after an @ sign), but you can combine a time with a date offset. For example, **/[d2010-12-08@12:00,+2@12:00]** selects files that were last modified between noon on December 8 and noon on December 10, 2010. Similarly, **/[d-2@15:00,+1]** selects files last modified between 3:00 pm the day before yesterday and the end of the day one day after that, *i.e.*, yesterday. The second time defaults to the end of the day because no time is specified.

You can exclude a date range by preceding the range with the ! character.

#### Notes:

- If the second date is the termination date, and it includes an explicit termination time, it is considered an exact value. For example, in the last example the termination time was 6PM. Files with a timestamp of 6:00:01 PM or later are not included in the date range. This is different from the behavior of [time ranges](#).
- If you include seconds in the times you specify, they will be silently ignored (no error or warning).
- If the first date is later than the second, any time of day modifiers for the first date are silently ignored.

#### Date types and selection

Windows file systems keep track of three dates for a file: when it was created, when it was last modified (written), and when it was last accessed. You specify which date and time is used in a date range by adding **a** (access), **c** (creation), or **w** (write) after the **d** in the range. For example, to select all files created between February 1, 2010 and February 7, 2010, inclusive, you would use **/[dc2010-02-1,2010-2-7]**. If you don't specify which date and time to use, **TCC** will use the date the file was last modified

(written).

**NOTE:** On FAT32 drives which support long filenames, only the last access date is recorded; the last access time is always returned as 00:00. However, on NTFS drives, last access information includes both date and time.

Date and time ranges may not always work as you expect across a network, including on FTP or HTTP servers, due to differences in time zone and file time storage method between the local and remote systems. Be sure to do some non-destructive testing before depending on date or time ranges to yield the results you want on a remote system.

### Defaults for Date Ranges

|                           |                                 |
|---------------------------|---------------------------------|
| Start date:               | Today                           |
| End date:                 | Today                           |
| Time of first parameter:  | Beginning of the day (00:00:00) |
| Time of second parameter: | End of the day (23:59:59)       |
| Missing second parameter: | Current date and time           |
| Date type                 | Modification (write)            |

#### 4.7.6.3 Time Ranges

Time ranges select files timed at any time between the two specified times of day. For example, to select files modified at or between noon and 2:00 PM on any day, use **/[t12:00p,2:00p]**. The times in a time range can either be in 12-hour format, with a trailing **a** for AM or **p** for PM, or in 24-hour format.

When you use a time range in a command it should immediately follow the command name. See [General Rules for Using Ranges](#) for additional details.

If you omit the second parameter in a time range, you will select files that were modified between the first time and the current time, on any date. You can also use offsets, beginning with a plus sign [**+**] or a minus sign [**-**] for either or both of the parameters in a time range. The offset values are interpreted as minutes. Some examples:

| Specification          | Selects Files                                                   |
|------------------------|-----------------------------------------------------------------|
| <b>/[t12:00p,+120]</b> | modified between noon and 2:00 PM on any date                   |
| <b>/[t-120,+120]</b>   | modified between two hours ago and the current time on any date |
| <b>/[t0:00,11:59]</b>  | modified in the morning on any date                             |

The separator character used in the time may vary depending upon your country information.

You can exclude a time range by preceding the range with the **!** character.

### Time types and selection

Windows keeps track of three times for a file: when it was created, when it was last modified (written), and when it was last accessed. You can specify which time is used in a time range by adding **a** (access), **c** (creation), or **w** (write) after the **t** in the range specification. For example, to select all files created between noon and 2:00 pm, you would use **/[tc12:00p,2:00p]**. If you don't specify which time to use, **TCC** will use the time the file was last modified (written).

**NOTE:** On FAT drives which support long filenames, only the last access date is recorded; the last access time is always returned as 00:00. However, on NTFS drives, last access information includes

both date and time.

Time ranges may not always work as you expect across a network, including on FTP or HTTP servers, due to differences in time zone and file time storage method between the local and remote systems. Be sure to do some non-destructive testing before depending on time ranges to yield the results you want on a remote system.

When you use a time range in a command it should immediately follow the command name. See [General Rules for Using Ranges](#) for additional details.

## Defaults

Start time: Current time  
End time: Current time  
Time type: Modification (last write)

### 4.7.6.4 File Exclusion

Most internal commands which accept wildcards also accept file exclusion ranges to further define the files that you wish to work with. **TCC** examines each file name and excludes files that match the names you have specified in the exclusion range.

When you use an exclusion range in a command it should immediately follow the command name. See [General Rules for Using Ranges](#) for additional details.

A file exclusion range begins with the switch character (usually a slash), followed by a left square bracket and an exclamation mark **!**. The range ends with a right square bracket **]**. You can specify multiple file exclusions (useful if you have a alias that is defining an exclusion and you want to pass another one as an argument).

Inside the brackets, you can list one or more filenames to be excluded from the command. The filenames can include [wildcards and extended wildcards](#), but may not include path names or drive letters. You can exclude directories by appending a **\** to the name.

The following example will display all files in the current directory except backup files (files with the extension **.BAK** or **.BK**):

```
dir /[!*.bak *.bk] *
```

You can combine file exclusion ranges with [date, time, and size ranges](#). This example displays all files that are 10K bytes or larger in size and that were created in the last 7 days, except **.C** and **.H** files:

```
dir /[s10k] /[d-7] /[!*.c *.h] *
```

File exclusion ranges, a unique feature of **TCC**, work for internal commands. The [EXCEPT](#) command can also be used to exclude files from processing by any external or internal command which ignores files with the hidden attribute. You can utilize the file exclusion range with external commands utilizing the [DO](#) or [FOR](#) command; however, the performance will not be as good, since the external command is started separately for each match.

**Note:** File exclusion first checks to see if a file specification with embedded brackets exactly matches an existing file. If no such file is found, it interprets the brackets as wildcards.

See also: [Include Lists](#).

#### 4.7.6.5 Owner Ranges

Most internal commands which accept wildcards also accept owner ranges to further define the files that you wish to work with. **TCC** examines each file or directory and excludes those whose owner doesn't match that in the exclusion range.

Owner ranges support wildcard comparisons. The value is the same as shown in [DIR /Q](#) or [%@owner](#).

The syntax is:

```
/[O"owner"]
```

If you precede the **O** with a **!**, the result is reversed.

The following example will display all files in the current directory owned by Bob:

```
dir /[O"*\Bob"] *
```

The following example will display all files in the current directory **except** those owned by Bob:

```
dir /![O"*\Bob"] *
```

#### 4.7.6.6 Description Ranges

Most internal commands which accept wildcards also accept description ranges to further define the files that you wish to work with.

When you use a description range in a command it should immediately follow the command name. See [General Rules for Using Ranges](#) for additional details.

A description range is specified as **/I"text"** where **text** is the description to be matched. [Wildcards](#) are supported. For example, **/I"\*agua\***" selects all files with the string **agua** somewhere in the file description. The search text must be enclosed in double quotes, and must immediately follow the **/I**, with no intervening spaces.

You can select all files that have a description with **/I"[?]\*"** (the **[?]** requires that the description contain at least one character, and the **\*** allows any text).

You can select all files that do not have a description with **/I"\\"** (the **\** requires that the first character, and therefore the descriptor itself, does not exist).

You can also search descriptions using [regular expressions](#) with **/R"text"**.

If you precede the **I** or **R** with a **!**, the result is reversed. For example, **!/I"\*beta\*"** will select all of the files that do **not** have the word **beta** in their description.

See [DESCRIBE](#) for details on file descriptions.

#### 4.7.7 Attribute Switches

Most file commands in **TCC** include the **/A:** switch, which allows you to select files for the command to process based on their [attributes](#). These switches all use the format **/A[:][+-]RHSAD**. The colon after **/A** is optional in [DIR](#), [FFIND](#), and [SELECT](#), but is required in all other commands. The characters after the **/A:** specify which attributes to select, as follows:

- R Read-only
- H Hidden
- S System
- A Archive
- D Directory

On NTFS volumes, the extended attributes below are also available.

- E Encrypted
- C Compressed
- I Not content-indexed
- L Symbolic link or Junction (reparse point)
- N Normal (cannot be used for file selection)
- O Offline
- P Sparse file
- T Temporary
- V Integrity (Windows 8 server ReFS only)
- X No scrub data (Windows 8 server ReFS only)

The **N** (normal) attribute is not stored on disk. It is dynamically generated by the operating system if none of the other attributes is set. Its use for file selection is not supported in either commands or variable functions.

If no attributes are listed at all (*i.e.*, **/A:**), the command will process all files, and (where applicable) all subdirectories, including hidden and system files and directories.

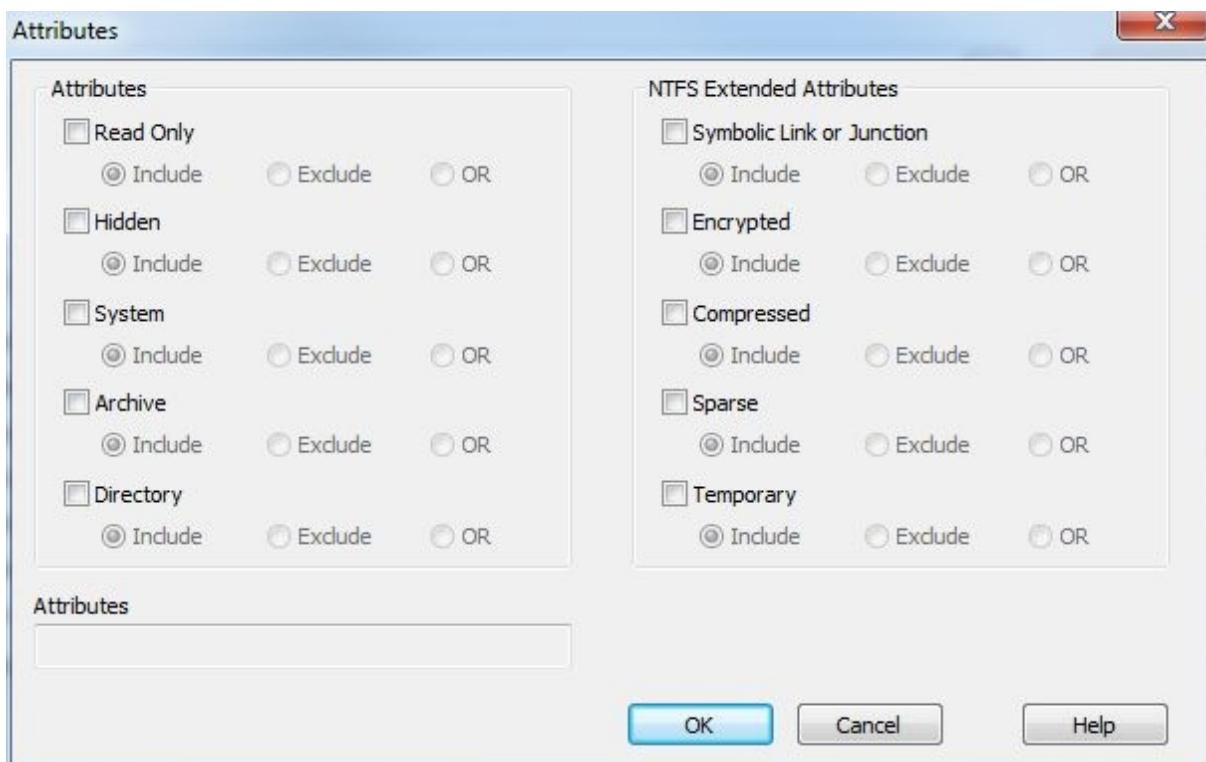
If attributes are combined, all the specified attributes must match for a file to be selected. For example, **/A:RHS** will select only those files with all three attributes set.

If you precede an attribute with a hyphen –, files with that attribute will be excluded. For example, **/A:RH-S** selects files which have the read-only and hidden attributes set and which do not have the system attribute set.

If you precede an attribute with a plus +, files will be selected which have that attribute turned on or off. When multiple attributes are preceded by +, only files which have at least one of these attributes will be selected. For example, **/A:+H+S** will select files with the hidden or system attribute, or both, but will not select files which have neither attribute set. **/A:R+H+S** will select files which are read-only, and also have the hidden or system attribute, or both.

You can combine the plus sign, hyphen, and unmarked attributes to build a specification as complex as you need.

If you use the format **/A:=**, TCC will display a dialog that allows you to select the attributes you want (not included in *TCC/LE*):



### Example

The (dangerous!) command below will make all hidden, system, and/or read-only files in the default directory visible and writeable, but not modify the attributes of files which are neither hidden nor system nor read-only (thus not reporting files already in the desired state):

```
attrib /e /p /a:+r+h+s -r -h -s
```

### 4.7.8 Multiple Filenames

Most file processing commands can work with multiple files at one time. To use multiple file names, you simply list the files one after another on the command line, separated by spaces. You can use [wildcards](#) in any or all of the filenames. For example, to copy all *.TXT* and *.DOC* files from the current directory to drive **A**, you could use this command:

```
copy *.txt *.doc a:
```

If the files you want to work with are not in the default directory, you must include the full path with each filename:

```
copy a:\details\file1.txt a:\details\file1.doc c:
```

Multiple filenames are handy when you want to work with a group of files which cannot be defined with a single filename and wildcards. They let you be very specific about which files you want to work with in a command.

When you use multiple filenames with a command that expects both a source and a destination, like [COPY](#) or [MOVE](#), be sure that you always include a specific destination on the command line. If you don't, the command will assume that the last filename is the destination and may overwrite important

files.

Like [extended wildcards](#) and [include lists](#), multiple filenames will work with internal commands but not with external programs, unless those programs have been written to handle multiple file names on the command line.

If you have a list of files to process that's too long to put on the command line or too time-consuming to type, see [@File Lists](#) as well as the [DO](#), [FOR](#) and [SELECT](#) commands for other ways of passing multiple file names to a command.

#### 4.7.9 Include Lists

Any internal command that accepts [multiple filenames](#) will also accept one or more include lists. An include list is simply a group of filenames, with or without wildcards, separated by semicolons [;]. Only the first entry in each include list may specify a path. All files in an include list must be in the same directory. You may not add a space on either side of the semicolon. See the rule below to determine when a [semicolon is part of a file name](#) and when it is an include list separator.

For example, you can shorten this command which uses multiple file names:

```
copy a:\details\file1.txt a:\details\file1.doc c:
```

to this using an include list:

```
copy a:\details\file1.txt;file1.doc c:
```

Include lists are similar to multiple filenames, but have three important differences.

- First, you don't have to repeat the path to your files if you use an include list, because all of the included files must be in the same directory.
- Second, if you use include lists, you aren't as likely to accidentally overwrite files if you forget a destination path for commands like [COPY](#), because the last name in the list will be part of the include list, and won't be seen as the destination file name. Include lists can only be used as the source parameter -- the location files are coming from -- for [COPY](#) and other similar commands. They cannot be used to specify a destination for files.
- Third, multiple filenames and include lists are processed differently by the [DIR](#) and [SELECT](#) commands. If you use multiple filenames, all of the files matching the first filename are processed, then all of the files matching the second name, and so on. When you use an include list, all files that match any entry in the include list are processed together, and will appear together in the directory display or [SELECT](#) list. You can see this difference clearly if you experiment with both techniques and the [DIR](#) command. For example,

```
dir \doc*.txt *.doc
```

will list all the *.TXT* files in directory *\DOC\* with a directory header, the file list, and a summary of the total number of files and bytes used. Then it will do the same for the *.DOC* files in the current directory. However,

```
dir \doc*.txt;*.doc
```

will display all the *.TXT* and *.DOC* files in directory *\DOC\* in one list.

Like [extended wildcards](#) and [multiple filenames](#), include lists work with internal commands, but not with external programs (unless they have been programmed especially to support them).

The maximum length of an include list is 32,767 characters (same as the maximum length of a single file name).

### Semicolons in filenames

Since a semicolon (";") is a valid (albeit unfortunate) character in a file name, you must quote any such name if you don't want **TCC** to treat it as an include list.

If a filename parameter includes a semicolon, **TCC** first attempts to find a filename containing an embedded semicolon. If found, that filename is used. If no file is found, the semicolon is considered to be an include list separator.

See also: [Exclusion Ranges](#).

## 4.7.10 Delayed Variable Expansion

Some of the internal commands ([COPY](#), [MOVE](#), [PDIR](#), [REN](#)) support delayed variable expansion for the target filename. The function argument must be an asterisk (\*), which will be replaced by the name of each matching source file. The variable function name must be preceded by two %%'s; the first one will be removed before the command is called, and the second when the command calls the variable expansion routine. This allows much greater flexibility in building the target filenames.

For example, to copy all of your \*.MP3 files, and append the string "\_saved" to the filename part :

```
copy *.mp3 %%@name [*]_saved.mp3
```

## 4.7.11 Extended Parent Directory Names

**TCC** has an extended syntax for referencing parent directories, by adding additional . characters. Each additional . represents an additional directory level above the current directory. For example, ..\FILE.DAT refers to a file in the current directory, ..\FILE.DAT refers to a file one level up, i.e., in the parent directory, and ...FILE.DAT refers to a file two levels up, i.e., in the parent of the parent directory. If your default directory is C:\DATA\FINANCE\JANUARY, you can copy the file LETTERS.DAT from directory C:\DATA to drive A: with the command

```
[C:\DATA\FINANCE\JANUARY] copy ... \LETTERS.DAT A:
```

**Note:** This extended notation may not be understood by external programs. Consider using the [@FULL](#) function to expand file and directory references when necessary:

```
[C:\DATA\FINANCE\JANUARY] myprog %@full[... \LETTERS.DAT]
```

## 4.7.12 LFN File Searches

There are some special considerations applicable to volumes which support long file names (including VFAT, FAT32, and NTFS volumes). All files on such volumes have a short (FAT-compatible 8.3) file name (SFN). A file which was created (or renamed to) a name which contains lower case letters or other characters not compatible with SFNs, or a name longer than 8 characters, or an extension longer than 3 characters, or more than one period(.) in its name will have both the [long file name](#) (LFN) specified, and an SFN automatically generated by the file system. The SFN associated with an LFN may change when

the file is moved or copied even when the LFN is not changed.

When CMD performs a wildcard search, it searches for both forms of each file name. The long filenames are checked first, followed by the short file names. Matching files which have only a short filename will be found during the first search, because in that case the file system treats the SFN name as if it were a LFN.

For example, suppose you have two files in a directory with these names:

| Long Name              | Short Name          |
|------------------------|---------------------|
| <i>Letter Home.DOC</i> | <i>LETTER~1.DOC</i> |
| <i>Letter02.DOC</i>    | <i>LETTER02.DOC</i> |

A search for *LETTER???.DOC* will find both files. The second file (*Letter02.DOC*) will be found during the search of long filenames. The first file (*Letter Home.DOC*) will be found during the search of short filenames but will return LFN.

Because this dual search can result in some very unexpected or even disastrous results, **TCC** defaults to searching only for the LFN. You can change the default with the **Search for SFNs** option in the OPTION / Startup dialog.

Take extra care when you use wildcards to perform operations on LFN volumes if you have set **Search for SFNs**, because you may select more files than you intended. For example, Windows often generates short filenames that end with **~1**, **~2**, etc. If you use a command such as:

```
del *1.*
```

you will delete all such files, including most files with long filenames, which is probably not the result you intended!

#### 4.7.13 @File Lists

Many internal commands allow you to specify a file containing a list of all of the files you want to process in the command line (instead of enumerating them individually). You specify that a file is a file list by prefixing its name with the @ sign, e.g., [LIST](#) @XXX specifies that [LIST](#) is to operate on the files listed in the file XXX instead of on XXX itself.

A file list is simply a standard text file containing the names of the files to process, one per line. This allows you to create a list of files for processing using output from [DIR /B](#), [DIR /F](#), or [FFIND](#), a text editor, or any other method that produces a file in the proper format. Both absolute and relative paths may be included in the file. However, wildcards are ignored, and each line is processed literally, without any further checking. This means that if a command allows options to restrict operations based on age (/U, /C), ranges (/I..., /[d..., /t...]), attributes (/A:), or location (/S), those restrictions will be ignored when processing the [@file](#) contents.

Commands supporting the [@File](#) syntax include:

|                             |                              |                       |
|-----------------------------|------------------------------|-----------------------|
| <a href="#">ATTRIB</a>      | <a href="#">FOR</a>          | <a href="#">TAIL</a>  |
| <a href="#">COPY</a>        | <a href="#">HEAD</a>         | <a href="#">TOUCH</a> |
| <a href="#">DEL / ERASE</a> | <a href="#">LIST</a>         | <a href="#">TYPE</a>  |
| <a href="#">DESCRIBE</a>    | <a href="#">MOVE</a>         | <a href="#">ZIP</a>   |
| <a href="#">DO</a>          | <a href="#">RD / RMDIR</a>   |                       |
| <a href="#">EXCEPT</a>      | <a href="#">REN / RENAME</a> |                       |

To use a file list, precede its name with an @ sign in the command. For example, to copy all of the files listed in *MYLIST.TXT* to *D:\SAVE\*:

```
copy @mylist.txt d:\save\
```

If you use a drive and/or path specification the @ sign can appear before the path or before the file name. For example, these are equivalent:

```
copy @e:\lists\mylist.txt d:\save\
copy e:\lists\@mylist.txt d:\save\
```

To use appropriately formatted data on the Windows clipboard as an catalog file use **@CLIP**: as the file name, for example:

```
copy @clip: d:\save\
```

### **@File Lists and "@" Signs in File Names**

Note that the @ sign is a rarely used, but legal filename character in Windows. If a file whose name begins with @ exists and you attempt to use an @file list with the same name, the file whose name begins with @ will take precedence. For example, if *C:\* contains both a file named *@MYLIST.TXT* and another named *MYLIST.TXT*, this command:

```
[c:\] copy @mylist.txt d:\save\
```

will copy the single file *@MYLIST.TXT* to *D:\SAVE\*, and will not process the list of files in *MYLIST.TXT*. To avoid this confusion, use a different name for one of the files.

#### **4.7.14 Switches for File Selection**

Many of the file processing commands ([ATTRIB](#), [COPY](#), [DEL](#), [DESCRIBE](#), [HEAD](#), [MOVE](#), [REN](#), [TAIL](#), [TYPE](#), etc.) support several standard switches for selecting files to process. Be sure to see the individual commands for details on which switches are supported for each command and how they work, and for additional switches specific to each command. Make sure that any [range](#) selections precede the options below in the command line.

The common file selection switches include:

- |                                    |                                                                                                                                                                                                                                                                |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>/A:[[-+]<br/>rhsadecijlopt]</b> | Select files based on their attributes, for example <b>/A:RH</b> selects files which have the read-only and hidden attributes set. See <a href="#">Attribute Switches</a> for details; see <a href="#">File Attributes</a> for more information on attributes. |
| <b>/N</b>                          | Don't actually process any files. This allows you to test what the results of a command would be, without actually performing the operation.                                                                                                                   |
| <b>/P</b>                          | Prompt for confirmation of each file.                                                                                                                                                                                                                          |
| <b>/S[n]</b>                       | Process files in the current directory and all of its subdirectories.                                                                                                                                                                                          |

## 4.8 Input / Output Redirection

This section covers features to change how **TCC** and some application programs handle input and output.

Internal commands and some external programs get their input from the computer's standard input device and send their output to the standard output device. Some programs, including **TCC**, also send special messages to the standard error device. Normally, the keyboard is used for standard input and the video display for both standard output and standard error, but you can temporarily change these assignments for special tasks.

For example, suppose you want a printed list of the files in a directory. If you change the standard output to the printer and issue a [DIR](#) command, the task is easy. DIR's output goes to the standard output device, and you have redirected standard output to the printer, so the DIR command prints filenames instead of displaying them on the screen. You can just as easily send the output of DIR (or any other command) to a file or a serial port.

We offer three methods of manipulating input and output: [Redirection](#), [Piping](#), and [Keystack](#). All three are explained in this section. In addition, **TCC** supports a subset of ANSI X3.64 control sequences in displayed text. The last topic in this section explains [ANSI X3.64 support](#) in detail.

Redirection and piping affect the standard input, standard output, and standard error devices. They do not work with application programs which read the keyboard hardware directly, or which write directly to the display. Because most Windows applications fall into that category, you will find that redirection and piping are most useful when they are combined with internal commands.

The [TEE](#) and [Y](#) commands are "pipe fittings" which add more flexibility to pipes.

**TCC**'s output is normally in ANSI. If you want to redirect output in Unicode, you need to either use the [/U startup option](#) in **TCC**, or the [Unicode Output](#) option in TCMD.INI.

- [Redirection and Piping](#)
- [ANSI X3.64 Support](#)
- [Keystack](#)
- [Page and File Prompts](#)

### 4.8.1 Redirection and Pipes

This section covers redirection and pipes. You can use these features to change how **TCC** and some application programs handle input and output.

Internal commands and some external programs get their input from the computer's standard input device and send their output to the standard output device. Some programs also send special messages to the standard error device. Normally, the keyboard is used for standard input and the video screen for both standard output and standard error, but you can temporarily change these assignments for special tasks.

For example, suppose you want a printed list of the files in a directory. If you change the standard output to the printer and issue a [DIR](#) command, the task is easy. DIR's output goes to the standard output device, and you have redirected standard output to the printer, so the [DIR](#) command prints filenames instead of displaying them on the screen. You can just as easily send the output of [DIR](#) (or any other command) to a file or a serial port.

Redirection and piping affect the standard input, standard output, and standard error devices. They do not work with application programs which read the keyboard hardware directly, or which write directly to the screen. Because most Windows applications fall into that category, you will find that redirection and piping are most useful when they are combined with internal commands.

The [TEE](#) and [Y](#) commands are "pipe fittings" which add more flexibility to pipes.

**TCC**'s output is normally in ANSI. If you want to redirect output in Unicode, you need to either use the [/U startup option](#) in **TCC**, or the [Unicode Output](#) option in TCMD.INI.

#### 4.8.1.1 Redirection

Redirection can be used to reassign the standard input (stdin), standard output (stdout), and standard error (stderr) devices from their default settings (the keyboard and screen) to another device such as NUL or serial port, to a file, or to the Windows clipboard. You must use some discretion when you use redirection with a device.

Redirection always applies to a specific command, and lasts only for the duration of that command. When the command is finished, the assignments for standard input, standard output, and standard error revert to whatever they were before the command.

**TCC**'s output is normally in ANSI. If you want to redirect output in Unicode, you need to either use the [/U startup option](#) in **TCC**, or the [Unicode Output](#) option in TCMD.INI.

In the descriptions below, **filename** means either the name of a file or of an appropriate device (**CON** for the keyboard and screen; **CLIP**: for the clipboard; **NUL** for the "null" device, etc.).

Here are the standard redirection options supported by **TCC** (see below for additional redirection options using numeric file handles):

- [Input redirection](#)
- [Output redirection](#)
- [Special considerations for specific commands](#)
- [NoClobber](#)
- [Multiple redirections](#)
- [Creating an empty file](#)
- [Redirection by handle](#)
- ["Here-document" redirection](#)
- ["Here-string" redirection](#)

#### Input redirection

`< filename` To get input from a file or device instead of from the keyboard.

#### Output redirection

|                                          | <b>overwrite</b>                    | <b>append</b>                       |
|------------------------------------------|-------------------------------------|-------------------------------------|
| standard output                          | <code>&gt; filename</code>          | <code>&gt;&gt; filename</code>      |
| standard error                           | <code>&gt;&amp;&gt; filename</code> | <code>&gt;&amp;&gt; filename</code> |
| merge standard output and standard error | <code>&gt;&amp; filename</code>     | <code>&gt;&amp; filename</code>     |

To use redirection, place the redirection symbol and **filename** at the end of the command line, after the command name and any parameters. For example, to redirect the output of the [DIR](#) command to a file

called *DIRLIST*, you could use a command line like this:

```
dir /b *.dat > dirlist
```

You can use any combination of input and output redirection for the same command, as appropriate for your purpose. For example, this command sends input to the external program **SORT** from the file *DIRLIST*, and sends output from **SORT** to the file *DIRLIST.SRT*:

```
sort < dirlist > dirlist.srt
```

You can redirect text to or from the Windows clipboard by using the pseudo-device name **CLIP**: (the colon is required).

If you redirect the output of a single internal command like **DIR**, the redirection ends automatically when that command is done. If you start a batch file with redirection, all of the batch file's output is redirected, and redirection ends when the batch file is done. Similarly, if you use redirection after the closing parenthesis of a [command group](#) (e.g., ...) > **report**, all of the output from the command group is redirected, and redirection ends when the command group is done.

### Special considerations for specific commands

You cannot redirect all output from the execution of a **DO** loop due to the restriction that the **DO** command and its matching **ENDDO** may not be part of a command group.

To redirect the output of a [TEXT](#) command, append the redirection syntax to the [TEXT](#) command.

When you execute a **FOR** or **GLOBAL** command, redirection is separately performed for each iteration, based on the directory current for that iteration. This can result in repeated overwriting of the output file, or the creation of a separate output file in each directory. To generate a single, cumulative output file, use [Command Grouping](#) as in the example below:

```
(for /r %f in (*.btm) echo %@full[%f]) > c:\temp\btmlst
```

### NoClobber

When output is directed to a file with **>**, **>&**, or **>&>**, and that file already exists, it will be overwritten. You can protect existing files by using the [SETDOS /N1](#) command, the **Protect redirected output files** setting on the [Startup tab](#) of the configuration dialogs, or the [Protect redirected output file](#) option.

When output is appended to a file with **>>**, **>>&**, or **>>&>**, the file will be created if it doesn't already exist. However, if the NoClobber mode is set as described above, append redirection will not create a new file; instead, if the output file does not exist a "File not found" or similar error will be displayed.

You can temporarily override the current setting of NoClobber by using an exclamation mark **[!]** after the redirection symbol. For example, to redirect the output of DIR to the file *DIROUT*, and allow overwriting of any existing file despite the NoClobber setting:

```
dir >! dirout
```

### Multiple redirections

Redirection is fully nestable. For example, you can invoke a batch file and redirect all of its output to a file or device. Output redirection on a command within the batch file will take effect for that command

only; when the command is completed, output will revert to the redirected output file or device in use for the batch file as a whole.

### Creating an empty file

You can use redirection to create an empty (zero-byte) file. To do so, enter `>filename` as a command, with no actual command before the `>` character. If you have enabled [Protect redirected output file](#), use `>!filename`.

### Redirection by handle

In addition to the redirection options above, **TCC** also supports the CMD syntax:

|                          |                                                               |
|--------------------------|---------------------------------------------------------------|
| <code>n&gt;file</code>   | Redirect handle <b>n</b> to the named file                    |
| <code>n&gt;&amp;m</code> | Redirect handle <b>n</b> to the same place as handle <b>m</b> |

**Warning:** You may not put any spaces between the **n** and the `>`, or between the `>`, `&`, and **m** in the second form. The values of **n** and **m** must be single decimal digits, and represent file handles. Windows defines **0**, **1**, and **2** as shown in the table below.

| Handle | Assignment      |
|--------|-----------------|
| 0      | standard input  |
| 1      | standard output |
| 2      | standard error  |

The `n>file` syntax redirects output from handle **n** to **file**. You can use this form to redirect two handles to different places. For example:

```
dir > outfile 2> errfile
```

sends normal output to a file called *OUTFILE* and any error messages to a file called *ERRFILE*.

The `n>&m` syntax redirects handle **n** to the same destination as the previously assigned handle **m**. For example, to send standard error to the same file as standard output, you could use this command:

```
dir > outfile 2>&1
```

Notice that you can perform the same operations by using standard redirection features. The two examples above could be written as

```
dir > outfile >&> errfile
```

and

```
dir >& outfile
```

### "Here-document" redirection

Wherever input redirection is supported, you can use a Linux-like "here-document" approach. The syntax is:

```
program << word
```

The current batch file is read up to the next occurrence of ***word***, and the resulting text becomes standard input to ***program***. For example:

```
c:\test\program.exe << endinput
input 1
input 2
input 3
endinput
echo This is the next line after "program.exe"
```

#### Special features of "here document":

- If the << is followed by a hyphen (-), the leading white space on the following lines will be removed before passing them to ***program*** (i.e. they will be effectively left-justified).
- The parser will perform variable expansion on each line, unless the word following << is enclosed in double quotes.

#### "Here-string" redirection (Not available in *TCC/LE*)

The "here-string" lets you send string text directly to a program's input. The syntax is:

```
program <<< string
```

This is similar to using [KEYSTACK](#), but easier to enter for text input. (If you need to send special keys or insert waits, you'll need to use KEYSTACK.) For example, to send a string to the standard input of program:

```
c:\test\program.exe <<< This is some input text.
```

#### 4.8.1.2 Pipes

Piping is a special form of redirection, using an additional instance of **TCC** for each instance of the ***piping*** specified in the command line.

You can create a ***pipe*** to send the *standard output* of a command (***command1***) to the *standard input* of another command (***command2***), and optionally also send the *standard error* as well:

| what is sent to pipe                     | command format                                   |
|------------------------------------------|--------------------------------------------------|
| standard output only                     | <b><i>command1</i></b>   <b><i>command2</i></b>  |
| merge standard output and standard error | <b><i>command1</i></b>  & <b><i>command2</i></b> |

For example, to take the output of the [ALIAS](#) command (which displays a list of your aliases and their values) and pipe it to the external SORT utility to generate a sorted list, you would use the command:

```
alias | sort
```

To do the same thing and then pipe the sorted list to the internal [LIST](#) command for full-screen viewing:

```
alias | sort | list /s
```

The [TEE](#) and [Y](#) commands are "pipe fittings" which add more flexibility to pipes.

**TCC**'s output is normally in ANSI. If you want to redirect output in Unicode, you need to either use the [/U startup option](#) in **TCC**, or the [Unicode Output](#) option in TCMD.INI.

Like redirection, pipes are fully nestable. For example, you can invoke a batch file and send all of its output to another command with a pipe. A pipe on a command within the batch file will take effect for that command only; when the command is completed, output will revert to the pipe in use for the batch file as a whole. You may also have 2 or more pipes operating simultaneously if, for example, you have the pipes running in different windows or processes.

### **Processing each line received from a pipe**

To process each line of text sent by the left side of a pipe in **TCC**, you may use the syntax below:

```
dir | for %file in (@CON:) command %file
```

This example shows how to pass each line of piped data to a **command**.

**WARNINGS:** **TCC** implements pipes by starting a new process for the receiving program. This process goes through the standard shell start-up procedure, including execution of the [TCSTART](#) file, for EACH receiving program. All of the sending and receiving programs run concurrently; the sending program writes to the pipe and the receiving program reads from the pipe. When the receiving program finds an End of File signal, it finishes reading and processing the piped data, and terminates. When you use pipes with **TCC**, make sure you consider the possible consequences from using a separate process to run the receiving program, especially that it cannot create/modify/delete environment variables of the sending program, and inclusion of a command to change directories in the [TCSTART](#) file may cause the new process to execute in a different directory. When you use more than one pipe in a single command, e.g. the second example above with [LIST](#), each pipe adds another instance of **TCC**. If you need to execute the pipe in the same context, use in-process pipes (see below).

### **In-Process Pipes (Not available in TCC/LE)**

In-process pipes work like the old-style DOS pipes, by creating a temporary output file, redirecting STDOUT to that file, and then redirecting the temp file to STDIN of the following command. The syntax is:

```
command1 !! command2
```

This is the same as doing:

```
command1 > temp.dat & command2 < temp.dat
```

but is easier to type & to read.

The advantage of in-process pipes is that *command2* will be run in the same context as *command1*, so you can do things like modify environment variables without having them discarded when *command2* exits. There are also some disadvantages to using this type of "pseudo-pipe" -- it will usually be slower than a true pipe; it will use some disk space for its temp file; and *command2* will not be started until *command1* has exited.

## 4.8.2 ANSI X3.64 Support

There is no support for ANSI X3.64 in Windows. For this reason, **TCC** contains its own limited ANSI X3.64 support (key substitutions are not supported, nor are double-width or double-height characters, or blinking characters). **TCC** interprets only its own output, not the output of external commands. In some cases you can redirect the output of an application program to a temporary file, then send it through **TCC** ANSI X3.64 interpreter, e.g., by using the [TYPE](#) command. This will display ANSI X3.64 correctly, but will not work with an interactive application.

To utilize the **TCC** built-in ANSI X3.64 support you must enable it from the [Windows tab](#) of the [configuration dialogs](#), or with the [SETDOS /A](#) command. You can determine whether or not ANSI X3.64 support is enabled with the [ANSI](#) internal variable.

Several commands in **TCC** provide alternatives for ANSI X3.64 commands. For example, there are commands to set the screen colors and display text in specific colors and locations. These commands are easier to understand and use than the ANSI X3.64 control sequences.

For information on the specific ANSI X3.64 commands supported by **TCC** see the [ANSI X3.64 Command Reference](#).

## 4.8.3 Keystack

The [KEYSTACK](#) command overcomes two weaknesses of input redirection:

- 1) some programs ignore standard input and read the keyboard through Windows APIs, and
- 2) input redirection doesn't end until the program or command terminates. You can't, for example, use redirection to send the first few commands to a program and then type the rest of the commands yourself. But [KEYSTACK](#) lets you do exactly that.

[KEYSTACK](#) sends keystrokes to an application program. Once the [KEYSTACK](#) buffer is empty, the program will receive the rest of its input from the keyboard. [KEYSTACK](#) is useful when you want a program to take certain actions automatically when it starts. It is most often used in batch files and aliases.

To place the letters, digits, and punctuation marks you would normally type for your program into the [KEYSTACK](#) buffer, enclose them in double quotes:

```
keystack "myfile"
```

Many other keys can be entered into the Keystack using their names. This example puts the **F1** key followed by the **Enter** key in the [KEYSTACK](#):

```
keystack F1 Enter
```

See [Keys and Key names](#) for details on how key names are entered. See the [KEYSTACK](#) command for information on using numeric key values along with or instead of key names, and other details about using the Keystack.

You must activate the window for the program that will receive the characters before you place them into the Keystack. See [KEYSTACK](#) for additional details; see [ACTIVATE](#) for information on activating a specific window.

## 4.8.4 Page and File Prompts

### Page Prompts

Several **TCC** commands can generate prompts, which wait for you to press a key to view a new page or to perform a file activity. When **TCC** is displaying information in page mode, for example with a [DIR /P](#) or [SET /P](#) command, it displays the message

Press ESC to quit, A to turn off paging or another key to continue...

At this prompt, you can press **Esc**, **Ctrl-C**, or **Ctrl-Break** if you want to quit the command. Pressing **A** will turn off the pause and prompt at the end of each page, and continue with the command. You can press almost any other key to continue with the command and see the next page of information.

### File Prompts

During file processing, if you have activated prompting with a command such as [DEL /P](#), you will see a prompt similar to the following before processing every file:

Y/N/A/R?

You can answer this prompt by pressing

|          |           |                                           |
|----------|-----------|-------------------------------------------|
| <b>Y</b> | Yes       | process this file                         |
| <b>N</b> | No        | do not process this file                  |
| <b>A</b> | All       | remaining files without further prompting |
| <b>R</b> | Remaining | files without further prompting           |

The **R** and **A** responses are equivalent; **A** was added for compatibility with CMD versions which display a **Yes/No/All** prompt. You can also press **Esc**, **Ctrl-C**, or **Ctrl-Break** at this prompt to cancel the remainder of the command.

If you press **Ctrl-C** or **Ctrl-Break** while a batch file is running, you will see a **Cancel batch job** prompt. For information on responses to this prompt see [Interrupting a Batch File](#).

## 4.9 Configuration

**TCC** offers a wide range of configuration options, allowing you to customize their operation for your needs and preferences. The **TCC OPTION** command invokes the [TCC Configuration Dialog](#).

We also discuss many ways of configuring **TCC** in other parts of the online help:

- With aliases and user-defined functions you can set default options for internal commands and create new commands (see [Aliases](#) and the [ALIAS](#) and [FUNCTION](#) commands).
- With [executable extensions](#) you can associate data files with the applications you use to open them.
- With the [FILECOMPLETION](#) environment variable or the [Filename Completion Options](#) configuration option, you can customize filename completion to match the command you are working with.
- With the [COLORDIR](#) environment variable or the [Directory Colors](#) configuration option you can set the colors used by the [DIR](#) command.

- With [command line options](#) you can specify where **TCC** looks for its startup files and how it operates for a specific instance.

## 4.9.1 Initialization Files

Part of the power of **TCC** is its flexibility, in allowing you to alter its configuration to match your style of computing. **TCC**'s configuration is controlled through a file of initialization information.

See [Locating the .INI files](#) below to find out how **TCC** locates its *TCMD.INI* file.

### Modifying the TCMD.INI File

You can create, add to, and modify the *TCMD.INI* file with the [configuration dialog](#), available via the [OPTION](#) command, or (if in a **Take Command** tab window), the **Configure TCC** entry in the [Options](#) menu.

Most of the changes you make in the [OPTION](#) command take effect immediately. A few (e.g., those associated with the startup screen size) only take effect when you start a new **TCC** session. See the online help for each individual dialog page if you are not sure when a change will take effect.

The dialogs handle most of the configuration options. The [Advanced directives](#) and the [Key Mapping directives](#) do not have corresponding fields in the configuration dialogs, and must be entered manually.

**TCC** reads its *TCMD.INI* file (see [Locating the .INI file](#)) when it starts, and configures itself accordingly. The *.INI* file is not reread when you change it manually. For manual changes to take effect, you must restart **TCC**.

Each item that you can include in the *.INI* file has a default value. You only need to include entries in the file for settings that you want to change from their default values.

The password fields in *TCMD.INI* (for example, the Internet password settings) are encrypted by the [OPTION](#) command.

### Using the TCMD.INI File

Some settings in the *.INI* file are initialized when you install **TCC**; others are modified as you use and when you exit **TCC**.

You can optionally include environment variables in the *TCMD.INI* [4NT] and [TCMD] sections; they will be expanded when *TCMD.INI* is loaded. If you want to delay expansion until command execution time (for example, with ColorDir) you will need to double the %'s.

### Locating the TCMD.INI File

#### 1) When starting **TCC** (a "primary shell"):

- If there is an `@d:\path\inifile` option on the startup command line, **TCC** will use the path and file name specified there.
- Otherwise, the default *.INI* file name in the table below is used, and the search starts in the directory where the **TCC** program file is stored. If the *.INI* file is not found, **TCC** will look in the %  
LOCALAPPDATA% directory.

If no *.INI* file is found, all options are set to their default values. A new *.INI* file will be created, using the default location and name, as explained above.

2) When starting **TCC** (a "secondary" shell) from another **TCC** shell:

**TCC** retrieves the primary shell's *.INI* file data, processes the [Secondary] section of the original *.INI* file if necessary, and then processes any **@d:\path\inifile** option on the secondary shell command line.

See [Command Line Options](#) for more details about the startup command line.

### TCMD.INI File Sections

The *TCMD.INI* file has a number of sections. Each section is identified by the section name in square brackets on a line by itself. **Take Command** stores the user-defined options in [**TakeCommand**]; **TCC** stores its user-defined options in [**4NT**].

The [**Primary**] and [**Secondary**] sections include directives that are used only in **TCC** primary and secondary shells, respectively. You don't need to set up these sections unless you want different directives for primary and secondary shells.

Directives in the [**Primary**] section are used for the first or primary shell. The values are passed automatically to all secondary shells, unless overridden by a directive with the same name in the [**Secondary**] section.

Directives in the [**Secondary**] section are used in secondary shells only, and override any corresponding primary shell settings.

Note that the terms **Primary** and **Secondary Shells** are now mostly obsolete in Windows. The interaction between Primary and Secondary **TCC** shells is limited to some minor inheritance (due to the design of Windows).

#### 4.9.1.1 Directives

This topic contains general information on **TCC** initialization. For information on specific directives see the separate topic for each type of directive:

- ▶ [Key Mapping Directives](#)
- ▶ [Advanced Directives](#)

These topics list the directives, with a short description of each, and a cross reference which selects a full description of that directive. A few of the directives are simple enough that the short description is sufficient, but in most cases you should check for any additional information in the cross reference topic if you are not already familiar with the directive.

#### Syntax for Directives

Most lines in the *.INI* file consist of a one-word **directive**, an equal sign =, and a **value**. For example, in the following line, the word **History** is the directive and **2048** is the value:

```
History = 2048
```

Any spaces before or after the equal sign are ignored.

Regardless of how long a string value is, for example the list for the [ColorDir](#) directive, you must enter it all on one line. Strings cannot be continued to a second line.

Each line must be within the [command line length limit](#).

The format of the **value** part of a directive line depends on the individual directive. It may be a numeric value, a single character, a choice (like **Yes** or **No**), a color setting, a key name, a path, a filename, or a text string. The value begins with the first non-blank character after the equal sign and ends at the end of the line or the beginning of a comment.

Blank lines are ignored in the *.INI* file and can be used to separate groups of directives.

You can place comments in the file by beginning a line with a semicolon ;. You can also place comments at the end of any line except one containing a text string value. To do so, enter at least one space or tab after the value, a semicolon, and your comment, like this:

```
History = 2048 ; set history list size
```

If you try to place a comment at the end of a string value, the comment will become part of the string and will probably cause an error.

If you use the [configuration dialogs](#) to modify the *.INI* file, comments on lines modified from within the dialogs will not be preserved when the new lines are saved. To be sure *.INI* file comments are preserved, put them on separate lines in the file.

When **Take Command** or **TCC** detects an error while processing the *.INI* file, it displays an error message and prompts you before processing the remainder of the file. This allows you to note any errors before the startup process continues. The directive in error will retain its previous or default value.

If you need to test different values for an *.INI* directive without repeatedly editing the *.INI* file, use the [OPTION](#) command or see the [INISet](#) directive.

The [SETDOS](#) command can override several of the *.INI* file directives. For example, the cursor shape used by **TCC** can be adjusted either with the CursorIns and CursorOver directives or the [SETDOS /S](#) command. The correspondence between a [SETDOS](#) option and a *.INI* directive is noted under both the individual help topic for that directive and under that option in the [SETDOS](#) help topic.

A **TCC** shell started from another **TCC** shell (a "secondary shell") automatically inherits the configuration settings currently in effect in the previous shell. If values have been changed by [SETDOS](#) or [OPTION](#) since the primary shell started, the current values will be passed to the secondary shell. If the previous shell's *.INI* file had a [Secondary] section, it will then be read and processed. If not, the previous shell's settings will remain in effect.

If you want to force secondary shells to start with a specific value for a particular directive, regardless of any changes made in a previous shell, repeat the directive in the [Secondary] section of the *.INI* file.

## Types of Directives

There are various types of directives in the *.INI* file. The type of a directive is shown under the individual help topic for that directive. The types are distinguished by the kind of data, if any, that must be entered after the = (equal sign):

- ▶ **Name = nnnn (1234)**: This directive takes a numeric value which replaces the "nnnn." The default value is shown in parentheses or listed below the directive's description.
- ▶ **Name = c (X)**: This directive accepts a single character as its value. The default character is shown in parentheses. You must type in the actual character; you cannot use a key name.
- ▶ **Name = CHOICE1 | Choice2 | ...** : This directive must be set to one of the vertical bar separated values listed between the braces. The default value is shown in all upper case letters in the directive description, but in your file any one of the choices can be entered, using any case. (Do not enter the vertical bar.) For example, if the choices were shown as YES | no then YES is the default.
- ▶ **Name = Color**: This directive takes a color specification. See [Colors and Color Names](#) for the format of color names.
- ▶ **Name = Key** : This directive takes a key specification. See [Keys and Keynames](#) for the format of key names.
- ▶ **Name = Path** : This directive takes a path specification, without a filename. The value should include both a drive and path (e.g., C:\TCMD) to avoid any possible ambiguities. A trailing backslash \ at the end of the path name is accepted but not required. Any default path is described in the text.
- ▶ **Name = File** : This directive takes a filename. We recommend that you use a full filename including the drive letter and path to avoid any possible ambiguities. Any default filename is described in the text.
- ▶ **Name = String** : This directive takes a string in the format shown. The text describes the default value and any additional requirements for formatting the string correctly. No comments are allowed.
- ▶ **Name** : This directive accepts NO parameters and the = is unnecessary (e.g. [ClearKeymap](#)).

## Evaluation of Directives

The directives are evaluated sequentially from top to bottom within each section processed. When a directive is processed more than once during startup, it replaces any previous value(s).

Most [key mapping](#) and [advanced](#) directives are cumulative and may appear several times when several concurrent values are desired, such as when assigning several different keystrokes to the same function.

### 4.9.1.1.1 Key Mapping Directives

These directives allow you to change the keys used for **TCC** command line editing and other internal functions. They cannot be entered via the [configuration dialogs](#); you must enter them manually (see the [.INI file](#) topic for details).

They are divided into four types, depending on the context in which the keys are used. For a discussion and list of directives for each type see:

- ▶ [General Input Keys](#)
- ▶ [Command Line Editing Keys](#)

- › [Popup Window Keys](#)
- › [LIST Keys](#)

Using a key mapping directive allows you to assign a different or additional key to perform the function described. For example, to use function key **F3** to invoke the [HELP](#) facility (normally invoked with **F1**):

```
Help = F3
```

Any directive can be used multiple times to assign multiple keys to the same function. For example:

```
ListFind = F ;F does a find in LIST
ListFind = F4 ;F4 also does a find in LIST
```

Use some care when you reassign keystrokes. If you assign a default key to a different function, it will no longer be available for its original use. For example, if you assign **F1** to the [AddFile](#) directive (a part of filename completion), the **F1** key will no longer invoke the help system, so you will probably want to assign a different key to Help.

See [Keys and Key Names](#) before using the key mapping directives.

Key assignments are processed before looking for keystroke aliases. For example, if you assign Shift-F1 to [HELP](#) and also assign Shift-F1 to a key alias, the key alias will be ignored.

Assigning a new keystroke for a function does not deassign the default keystroke for the same function. If you want to deassign one of the default keys, use the [NormalKey](#), [NormalEditKey](#), [NormalPopupKey](#) or [NormalListKey](#) directive. You must also deassign default keys before you can assign them to a different usage.

**Note:** if you assign the same key to two different functions, the first assignment found in the list will be used.

#### 4.9.1.1.1 General Input Keys

These directives apply to all input. They are in effect whenever **TCC** requests input from the keyboard, including during [command line editing](#) and the [DESCRIBE](#), [ESET](#), [INPUT](#), [LIST](#), and [SELECT](#) commands. The general input keys are:

|                                |                                                  |
|--------------------------------|--------------------------------------------------|
| <a href="#">Backspace</a>      | Deletes the character to the left of the cursor  |
| <a href="#">BeginLine</a>      | Moves the cursor to the start of the line        |
| <a href="#">Copy</a>           | Copies highlighted text to the clipboard         |
| <a href="#">Del</a>            | Deletes the character at the cursor              |
| <a href="#">DelToBeginning</a> | Deletes from the cursor to the start of the line |
| <a href="#">DelToEnd</a>       | Deletes from the cursor to the end of the line   |
| <a href="#">DelWordLeft</a>    | Deletes the word to the left of the cursor       |
| <a href="#">DelWordRight</a>   | Deletes the word to the right of the cursor      |
| <a href="#">Down</a>           | Moves the cursor or scrolls the display down     |
| <a href="#">EndLine</a>        | Moves the cursor to the end of the line          |
| <a href="#">EraseLine</a>      | Deletes the entire line                          |
| <a href="#">ExecLine</a>       | Executes or accepts a line                       |
| <a href="#">Ins</a>            | Toggles insert / overstrike mode                 |
| <a href="#">Left</a>           | Moves the cursor or scrolls the display left     |
| <a href="#">NormalKey</a>      | Deassigns a key                                  |

|           |                                               |
|-----------|-----------------------------------------------|
| Paste     | Pastes line from clipboard                    |
| Right     | Moves the cursor or scrolls the display right |
| Up        | Moves the cursor or scrolls the display up    |
| WordLeft  | Moves the cursor left one word                |
| WordRight | Moves the cursor right one word               |

**Backspace = Key**

**Default:** Bksp

Adds **Key** to the list of keys available during command line entry to delete the character to the left of the cursor.

See other [General Input Keys](#).

**BeginLine = Key**

**Default:** Home

Specifies **key** during command line entry as a request to move the cursor to the beginning of the line.

See other [General Input Keys](#).

**Copy = Key**

**Default:** Ctrl-Y

Adds **Key** to the list of keys available during command line entry to copy the highlighted text to the clipboard.

See other [General Input Keys](#).

**Del = Key**

**Default:** Del

Deletes the character at the cursor.

See other [General Input Keys](#).

**DelToBeginning = Key**

**Default:** Ctrl-Home

Deletes from the cursor to the start of the line.

See other [General Input Keys](#).

**DelToEnd = Key**

**Default:** Ctrl-End

Deletes from the cursor to the end of the line.

See other [General Input Keys](#).

**DelWordLeft = Key**

**Default:** Ctrl-L

Deletes the word to the left of the cursor.

See other [General Input Keys](#).

**DelWordRight = Key**

**Default:** Ctrl-R, Ctrl-Bksp

Deletes the word to the right of the cursor. See [ClearKeyMap](#) if you need to remove the default mapping of Ctrl-Bksp to this function.

See other [General Input Keys](#).

**Down = Key**

**Default:** Down

Scrolls the display down one line in [LIST](#); moves the cursor down one line in [SELECT](#) and in the command line history, directory history, or [@SELECT](#) window.

See other [General Input Keys](#).

**EndLine = Key**

**Default:** End

Moves the cursor to the end of the line.

See other [General Input Keys](#).

**EraseLine = Key**

**Default:** Esc

Deletes the entire line.

See other [General Input Keys](#).

**ExecLine = Key**

**Default:** Enter

Executes or accepts a line.

See other [General Input Keys](#).

#### **Ins = Key**

**Default:** Ins

Toggles insert / overstrike mode during line editing.

See other [General Input Keys](#).

#### **Left = Key**

**Default:** left arrow, ←

Specifies a key, such the using the key will move the cursor left.

See other [General Input Keys](#).

#### **NormalKey = Key**

Deassigns a general input key in order to disable the usual meaning of the key and/or make it available for keystroke aliases. This will make the keystroke operate as a "normal" key with no special function. For example:

```
NormalKey = Ctrl-End
```

will disable Ctrl-End, which is the standard "delete to end of line" key. Ctrl-End could then be assigned to a keystroke alias. Another key could be assigned the "delete to end of line" function with the [DelToEnd](#) directive.

See other [General Input Keys](#).

#### **Paste = Key**

**Default:** Ctrl-V

Paste the first line of the clipboard to the input line at the cursor position.

See other [General Input Keys](#).

#### **Right = Key**

**Default:** Right

Moves the cursor right one character on the input line; scrolls the display right 8 columns in [LIST](#); scrolls the display right 4 columns in the command line history, directory history, or [@SELECT](#) window.

See other [General Input Keys](#).

**Up** = Key

**Default:** Up

Scrolls the display up one line in [LIST](#); moves the cursor up one line in [SELECT](#) and in the command line history, directory history, or [@SELECT](#) window.

See other [General Input Keys](#).

**WordLeft** = Key

**Default:** Ctrl-Left

Moves the cursor left one word; scrolls the display left 40 columns in [LIST](#).

See other [General Input Keys](#).

**WordRight** = Key

**Default:** Ctrl-Right

Moves the cursor right one word; scrolls the display right 40 columns in [LIST](#).

See other [General Input Keys](#).

#### 4.9.1.1.2 Command Line Editing Keys

These directives apply only to [TCC command line editing](#). They are only effective at the prompt. The command line editing keys are:

|                               |                                                           |
|-------------------------------|-----------------------------------------------------------|
| <a href="#">AddFile</a>       | Keeps filename completion entry and adds another          |
| <a href="#">AliasExpand</a>   | Expands aliases on the command line                       |
| <a href="#">CommandEscape</a> | Allows direct entry of a keystroke                        |
| <a href="#">DelHistory</a>    | Deletes a history list entry                              |
| <a href="#">DirWinOpen</a>    | Opens the directory history window                        |
| <a href="#">EndHistory</a>    | Displays the last entry in the history list               |
| <a href="#">Help</a>          | Invokes this help system                                  |
| <a href="#">HelpWord</a>      | Invokes help for the word at the cursor                   |
| <a href="#">HistWinOpen</a>   | Opens the command history window                          |
| <a href="#">LastHistory</a>   | Recall the last history entry                             |
| <a href="#">LFNToggle</a>     | Toggles between long and short filenames                  |
| <a href="#">LineToEnd</a>     | Copies a line to the end of the history, then executes it |
| <a href="#">NextFile</a>      | Gets the next matching filename                           |
| <a href="#">NextHistory</a>   | Recalls the next command from the history                 |
| <a href="#">NormalEditKey</a> | Deassigns a command line editing key                      |
| <a href="#">PopFile</a>       | Opens the filename completion window                      |
| <a href="#">PrevFile</a>      | Gets the previous matching filename                       |
| <a href="#">PrevHistory</a>   | Recalls the previous command from the history             |

|                                |                                                   |
|--------------------------------|---------------------------------------------------|
| <a href="#">Redo</a>           | Redo the previous undo                            |
| <a href="#">RepeatFile</a>     | Repeats previous match during filename completion |
| <a href="#">SaveHistory</a>    | Saves the command line without executing it       |
| <a href="#">Undo</a>           | Undo the last edit                                |
| <a href="#">VariableExpand</a> | Expand variables on the command line              |

**AddFile = Key****Default:** F10

Adds **Key** to the list of keys available during command line entry to keep the current filename completion entry and insert the next matching filename.

See other [Command Line Editing Keys](#).

**AliasExpand = Key****Default:** Ctrl-F

Adds **Key** to the list of keys available during command line entry to expand all aliases in the current command line without executing them.

See other [Command Line Editing Keys](#).

**CommandEscape = Key****Default:** Alt-255

Adds **Key** to the list of keys available during command line entry to signify that the immediately subsequent keystroke is to be used literally, and not for command line editing control.

See other [Command Line Editing Keys](#).

**DelHistory = Key****Default:** Ctrl-D

Deletes the displayed history list entry and displays the previous entry.

See other [Command Line Editing Keys](#).

**DirWinOpen = Key****Default:** Ctrl-PgUp, F6

Opens the directory history window while at the command line.

See other [Popup Window Keys](#).

**EndHistory = Key****Default:** Ctrl-E

Displays the last entry in the history list.

See other [Command Line Editing Keys](#).

#### **Help = Key**

**Default:** F1

Displays the [Help File](#) topic for the current command. See also: the [HELP](#) command and the [HelpWord](#) directive.

See other [Command Line Editing Keys](#).

#### **HelpWord = Key**

**Default:** Ctrl-F1

Invokes the [HELP](#) facility for the word at the cursor.

See other [Command Line Editing Keys](#).

#### **HistWinOpen = Key**

**Default:** PgUp

Brings up the history window while at the command line.

See other [Popup Window Keys](#).

#### **LastHistory = Key**

**Default:** F3

Returns the last history entry. (Mostly useless; it is for compatibility with CMD.)

See other [Command Line Editing Keys](#).

#### **LFNToggle = Key**

**Default:** Ctrl-A

Toggles filename completion between long filename and short filename modes on LFN drives.

See other [Command Line Editing Keys](#).

#### **LineToEnd = Key**

**Default:** Ctrl-Enter

Copies the current command line to the end of the history list, then executes it.

See other [Command Line Editing Keys](#).

#### **NextFile = Key**

**Default:** F9, Tab

Gets the next matching filename during filename completion. See [ClearKeyMap](#) if you need to remove the default mapping of Tab to this function.

See other [Command Line Editing Keys](#).

#### **NextHistory = Key**

**Default:** Down

Recalls the next command from the command history.

See other [Command Line Editing Keys](#).

#### **NormalEditKey = Key**

Deassigns a command line editing key in order to disable the usual meaning of the key while editing a command line, and/or make it available for keystroke aliases. This will make the keystroke operate as a "normal" key with no special function. See [NormalKey](#) for an example.

#### **PopFile = Key**

**Default:** F7, Ctrl-Tab

Opens the filename completion window. Note that **Take Command** uses Ctrl-Tab to select windows. See [ClearKeyMap](#) if you need to remove the default mapping of Ctrl-Tab to this function.

See other [Command Line Editing Keys](#).

#### **PrevArgument = Key**

**Default:** Ctrl-B

Recall the last argument from the previous command line.

See other [Command Line Editing Keys](#).

#### **PrevFile = Key**

**Default:** F8, Shift-Tab

Gets the previous matching filename. See [ClearKeyMap](#) if you need to remove the default mapping of Shift-Tab to this function.

See other [Command Line Editing Keys](#).

**PrevHistory = Key**

**Default:** Up

Recalls the previous command from the command history.

See other [Command Line Editing Keys](#).

**Redo = Key**

**Default:** Alt-Y

Redo the last undo in the TCC command input.

See other [Command Line Editing Keys](#).

**RepeatFile = Key**

**Default:** F12

Repeats the previous matching filename during filename completion.

See other [Command Line Editing Keys](#).

**SaveHistory = Key**

**Default:** Ctrl-K

Saves the command line in the command history list without executing it.

See other [Command Line Editing Keys](#).

**Undo = Key**

**Default:** Alt-Z

Undo the last edit in the TCC command input.

See other [Command Line Editing Keys](#).

**VariableExpand = Key**

**Default:** Ctrl-X

Expands variables at the command prompt.

See other [Command Line Editing Keys](#).

## 4.9.1.1.3 LIST Keys

These directives are effective only inside the **TCC LIST** command.

|               |                                                         |
|---------------|---------------------------------------------------------|
| ListBack      | Return to the previous file                             |
| ListClipboard | Copy the current filename to the clipboard              |
| ListContinue  | Continue to the next file                               |
| ListExit      | Exits the current file                                  |
| ListFind      | Prompts and searches for a string                       |
| ListFindRegex | Prompt and search for a regular expression              |
| ListHex       | Toggles between hexadecimal and character display modes |
| ListHighBit   | Toggles LIST's "strip high bit" option                  |
| ListInfo      | Displays information about the current file             |
| ListNext      | Finds the next matching string                          |
| ListOpen      | Displays the "open file" dialog                         |
| ListPrevious  | Finds the previous matching string                      |
| ListPrint     | Prints the file on the default printer                  |
| ListRefresh   | Refresh the display                                     |
| ListUnicode   | Toggles Unicode display mode                            |
| ListWrap      | Toggles LIST's wrap option                              |
| NormalListKey | Deassigns a LIST key                                    |

**ListBack = Key**

**Default:** B

Returns to the previous file.

See other [LIST Keys](#).

**ListClipboard = Key**

**Default:** Ctrl-B

Copy the current [LIST](#) filename to the clipboard

See other [LIST Keys](#).

**ListContinue = Key**

**Default:** C

Go to the next file.

See other [LIST Keys](#).

**ListExit = Key**

**Default:** Esc

Exits from the [LIST](#) command.

See other [LIST Keys](#).

**ListFind = Key**

**Default:** F

Prompts and searches for a string.

See other [LIST Keys](#).

**ListFindRegex = Key**

**Default:** R

Perform a regular expression search in [LIST](#).

See other [LIST Keys](#).

**ListHex = Key**

**Default:** X

Toggles between hexadecimal and character display modes.

See other [LIST Keys](#).

**ListHighBit = Key**

**Default:** H

Toggles LIST's "strip high bit" option, which can aid in displaying files from certain word processors.

See other [LIST Keys](#).

**ListInfo = Key**

**Default:** I

Displays information about the current file.

See other [LIST Keys](#).

**ListNext = Key**

**Default:** N

Finds the next matching string.

See other [LIST Keys](#).

**ListOpen = Key**

**Default:** O

Opens the common Windows "open file" dialog to select a new file to [LIST](#).

See other [LIST Keys](#).

**ListPrevious = Key**

**Default:** Ctrl-B

Finds the previous matching string.

See other [LIST Keys](#).

**ListPrint = Key**

**Default:** P

Prints the file on the default printer.

See other [LIST Keys](#).

**ListRefresh = Key**

**Default:** F5

Refresh the [LIST](#) display. (Useful when viewing a growing log file.)

See other [LIST Keys](#).

**ListUnicode = Key**

**Default:** U

Toggles the [LIST](#) display mode between Unicode and ASCII.

See other [LIST Keys](#).

**ListWrap = Key**

**Default:** W

Toggles [LIST](#)'s wrap option on and off. The wrap option wraps text at the right margin.

See other [LIST Keys](#).

**NormalListKey = Key**

Deassigns a [LIST](#) key in order to disable the usual meaning of the key within LIST. This will make the keystroke operate as a "normal" key with no special function. See [NormalKey](#) for an example.

See other [LIST Keys](#).

#### 4.9.1.1.4 Popup Window Keys

The following directives apply to popup windows, including the command history window, the directory history window, the filename completion window, the extended directory search window, and the [@SELECT](#) window.

|                                |                                             |
|--------------------------------|---------------------------------------------|
| <a href="#">NormalPopupKey</a> | Deassigns a popup window key                |
| <a href="#">PopupWinDel</a>    | Deletes a line from within the popup window |

|                                 |                                                      |
|---------------------------------|------------------------------------------------------|
| <a href="#">PopupWinEdit</a>    | Moves a line from the popup window to the prompt     |
| <a href="#">PopupWinEditWin</a> | Edit a line in the popup window                      |
| <a href="#">PopupWinExec</a>    | Selects the current item and closes the popup window |

**NormalPopupKey = Key**

Deassigns a popup window key in order to disable the usual meaning of the key within the popup window. This will make the keystroke operate as a "normal" key with no special function. See [NormalKey](#) for an example.

See other [Popup Window Keys](#).

**PopupWinDel = Key**

**Default:** Ctrl-D

Deletes a line from within the command history or directory history window.

See other [Popup Window Keys](#).

**PopupWinEdit = Key**

**Default:** Ctrl-Enter

Moves a line from the command history or directory history window to the prompt for editing.

See other [Popup Window Keys](#).

**PopupWinEdit = Key**

**Default:** Ctrl-E

Edit a line in the command history or directory history window.

See other [Popup Window Keys](#).

**PopupWinExec = Key**

**Default:** Enter

Selects the current item and closes the window.

See other [Popup Window Keys](#).

#### 4.9.1.1.2 Advanced Directives

These directives are generally used for unusual circumstances, or for diagnosing problems. Most often they are not needed in normal use. They cannot be entered via the [configuration dialogs](#); you must enter them manually (see the [.INI file](#) for details).

|                              |                                                 |
|------------------------------|-------------------------------------------------|
| <a href="#">AliasSize</a>    | Set the global alias list size                  |
| <a href="#">AutoFirewall</a> | Enable / disable automatic firewall detection   |
| <a href="#">AutoProxy</a>    | Enable / disable automatic HTTP proxy detection |
| <a href="#">ClearKeyMap</a>  | Clear default key mappings                      |

|                    |                                                           |
|--------------------|-----------------------------------------------------------|
| CMDBatchDelimiters | Use CMD delimiter characters in batch file commands       |
| CMDVariables       | Use the CMD variable syntax (%var%)                       |
| CompleteAllFiles   | Match all extensions at the beginning of a command line   |
| Copyright          | Enable / disable the TCC copyright message display        |
| Debug              | Set debugging options                                     |
| DelWipePasses      | Set default passes for DEL /W                             |
| EverythingSearch   | Use Everything Search for CDD fuzzy directory searches    |
| FilesCaseSensitive | Enable / disable case sensitivity in filename comparisons |
| FunctionSize       | Set the global user-defined functions list size           |
| INIQuery           | Query for each line in the .INI file                      |
| LanguageDLL        | Set localized language DLL                                |
| MSAAMenu           | Check for screen readers & disable detachable menus       |
| NoINIErrors        | Don't display error messages for TCMD.INI errors          |
| StartTabWait       | Delay after starting each TCMD tab                        |
| TrayHotKey         | Hotkey to toggle TCMD to / from the system tray           |
| UpdateINI          | Enable / disable changes to the .INI file.                |
| UTF8Output         | Write TCC redirected output in UTF-8 format               |

#### 4.9.1.1.2.1 AliasSize

**AliasSize = n**

AliasSize allows you to set the size of the global alias list (in characters). The default is 256K; you should only need to change this if you have an exceptionally large alias list or if you want to minimize **TCC**'s memory footprint.

You do not need to set the alias list size if you're using local lists -- **TCC** will automatically resize the alias list as needed.

#### 4.9.1.1.2.2 AutoFirewall

**AutoFirewall=YES|no**

If set to "yes", **TCC** will attempt to automatically detect and use HTTP firewall system settings (if available).

#### 4.9.1.1.2.3 AutoProxy

**AutoProxy=YES|no**

If set to "yes", **TCC** will attempt to automatically detect and use HTTP proxy server system settings (if available).

#### 4.9.1.1.2.4 ClearKeyMap

**ClearKeyMap**

Clears all current [key mappings](#). ClearKeyMap is a special directive which has no value or = after it. Use ClearKeyMap with caution - it deletes all of the default definitions, and also any definitions in your .INI file directives that are processed before ClearKeyMap is processed. It is useful only if you want to make available most of the keys which have default assignments for other purposes, e.g., for keystroke

aliases. ClearKeyMap should appear before any other key mapping directives. You may restore default mappings to keys you want to retain using the appropriate key assignment directives, e.g., [NextFile=Tab](#).

To clear the default mappings for just a few keys, use the [NormalEditKey](#), [NormalKey](#), [NormalListKey](#), and/or [NormalPopupKey](#) directives.

See other [Advanced Directives](#).

#### 4.9.1.1.2.5 CMDBatchDelimiters

**CMDBatchDelimiters**=YES|no

If set to "no", TCC will not treat an = as a batch argument delimiter. (Note: this will break CMD compatibility!). This is only for users with (very) old 4NT scripts; it is strongly discouraged for new installations.

#### 4.9.1.1.2.6 CMDVariables

**CMDVariables** = yes | NO

TCC allows you to specify variables with only a single leading %; CMD requires both a leading and a trailing %. Normally TCC is able to detect this, but if you have variable names with embedded whitespace (a bad idea!) TCC will not expand the variable.

If you need as close to 100% CMD compatibility as possible, and you don't care about running existing TCC batch files or aliases, setting **CMDVariables=yes** will allow TCC to properly expand these types of variable names.

#### 4.9.1.1.2.7 CompleteAllFiles

**CompleteAllFiles** = yes | NO

Normally, **TCC** will only complete directories and executable files (as defined by PATHEXT) when you press Tab or F9 at the beginning of a command line. If CompleteAllFiles is set to YES, **TCC** will complete any matching filename. Note that if you also have CompletePaths set, you'll probably have several hundred (or thousand!) matches for any filename you enter.

#### 4.9.1.1.2.8 ConsolePopupWindows

**ConsolePopupWindows** = yes | NO

Enable or disable character-mode popup windows (for example, command or directory history windows). This is intended for use with server consoles that are character-mode only, or when using SSH with no GUI support. There is no benefit (and several disadvantages) in using this option for normal non-server environments.

#### 4.9.1.1.2.9 Copyright

**Copyright**=YES | no

Display the TCC copyright message at startup. This is the same as the TCC /Q startup option, and only applies to registered copies.

## 4.9.1.1.2.10 Debug

**Debug = n****Default:** 2

Controls certain debugging options which can assist you in tracking down unusual problems. Use the following values for **Debug** (to enable more than one option, add the desired values together):

- 0 Disabled.
- 1 During the startup process, display the complete command tail passed to **Take Command**, then wait for a keystroke.
- 2 (default) Include the product name with every error message displayed by **Take Command**. This may be useful if you are unsure of the origin of a particular error message.

See also: the [batch file debugger](#), a separate and unrelated facility for stepping through batch files.

See other [Advanced Directives](#).

## 4.9.1.1.2.11 DelWipePasses

**DelWipePasses = n****Default:** 3

Sets the default number of passes for a DEL /W (wipe). If you have a slow disk drive you might want to set this to 1 or 2. The range is 1-999 (but there's little reason to set it higher than 3 or 4).

## 4.9.1.1.2.12 EverythingSearch

**EverythingSearch = yes | NO**

If YES, CD and CDD will use "Everything Search" (<http://www.voidtools.com>) instead of JPSTREE.IDX for fuzzy directory matching. See [CDD](#) for details.

## 4.9.1.1.2.13 FileCompletionLooping

**FileCompletionLooping = yes | NO**

Enable or disable Linux-style filename completion looping. I.e., when **TCC** reaches the last match, it will loop back to the first match (with no indication that it has done so).

It's easier to use the tab / F8 forward/back stepping in **TCC**.

## 4.9.1.1.2.14 FilesCaseSensitive

**FilesCaseSensitive = yes | NO**

If YES, filename comparisons will be case sensitive (like Linux, and unlike Windows). This will only affect filename matching within TCC.

## 4.9.1.1.2.15 FunctionSize

**FunctionSize = n**

FunctionSize allows you to set the size of the global user-defined function list (in characters). The default is 128K; you should only need to change this if you have an exceptionally large function list or if you want to minimize **TCC**'s memory footprint.

You do not need to set the function list size if you're using local lists -- **TCC** will automatically resize the function list as needed.

#### 4.9.1.1.2.16 INIQuery

**INIQuery** = yes | NO

If set to **Yes**, a prompt will be displayed before execution of each subsequent line in the current .INI file. This allows you to modify certain directives when you start **Take Command** or **TCC** in order to test different configurations. INIQuery can be reset to **No** at any point in the file. Normally INIQuery = Yes is only used during testing of other .INI file directives.

The dialog displayed when INIQuery = Yes gives you three options:

**Yes** Executes the directive

**No** Skips the directive

**Cancel** Executes the directive and all remaining directives in the [TakeCommand] section of the .INI file (i.e., cancels the INIQuery = Yes setting)

See other [Advanced Directives](#).

#### 4.9.1.1.2.17 LanguageDLL

**LanguageDLL** = *filename*

Specifies the filename of the language DLL **Take Command** and **TCC** should use (English.dll, French.dll, German.dll, Italian.dll, Russian.dll, or Spanish.dll). **Take Command** normally uses the language dll that matches the default Windows user language, but you can override it with this directive.

(In most cases you shouldn't set this directive -- if you use a non-default language dll, you will get a mix of one language from **Take Command** and another from Windows for the system error messages.)

See other [Advanced Directives](#).

#### 4.9.1.1.2.18 Lua

**Lua** = YES | no

Enable or disable executing Lua scripts directly from the **TCC** prompt.

#### 4.9.1.1.2.19 MSAAMenu

**MSAAMenu** = yes | NO

**Take Command** checks to see if a screen reader is installed, and if so it sets MSAAMenu=Yes. This is to avoid problems with the **Take Command** detachable menus and some screen readers.

#### 4.9.1.1.2.20 NoNIErrors

**NoNIErrors** = yes | NO

If set to YES, no TCMD.INI parsing error messages will be displayed for all lines following the NoINIErrors line. This directive would normally be placed at the beginning of the [TakeCommand] and/or [4NT] section. It will **not** apply to the other section, so if you want all parsing errors suppressed you need to add NoINIErrors=Yes to both sections.

Note that setting this directive is generally **not** recommended, as it will suppress potentially critical errors.

#### 4.9.1.1.2.21 StartTabWait

##### **StartTabWait = n**

The number of milliseconds to wait between launching each *Take Command* startup tab. The range is 0 (default) to 5000.

This should only be needed in rare cases when tabs are interfering with one another while starting (for example, in their TCSTART code).

#### 4.9.1.1.2.22 TrayHotKey

##### **TrayHotKey = Z**

The hotkey to toggle Take Command to and from the system tray. The specified alphabetic key is combined with Ctrl + Shift, so the default hotkey is Ctrl-Shift-Z.

#### 4.9.1.1.2.23 UpdateINI

##### **UpdateINI = YES | no**

Enable or disable changes to the .INI file. (Useful for administrators who want to prevent users from changing their configuration.)

#### 4.9.1.1.2.24 UTF8Output

**UTF8Output = yes | NO** : The **TCC** output files (such as redirected output and pipes) will be written in UTF-8 format.

## 4.9.2 Configuration Dialog

This dialog, available via the [OPTION](#) command, contains several "pages" or "tabs" of options that let you change the way **TCC** looks and works.

The configuration dialog displays the name of the active *TCMD.INI* file in the title bar.

Unless you select the **Cancel** button, any changes you make will take effect immediately. If you select **Apply**, the settings will only apply for the duration of that session. If you select **OK**, the settings will be recorded in the appropriate main section (**[4NT]**) of the *TCMD.INI* file and will be in effect each time you start that command processor. If you want to set configuration directives in the **[Primary]** or **[Secondary]** sections of the *.INI* file, you must edit the file directly instead of using the dialog. Similarly, if you modified directives that originally resided in a separate included INI file, new directives will be saved to the main *.INI* file but the included file itself will not be altered. It would be wise to verify that no "old" directive in included files override the changes you made into the main file.

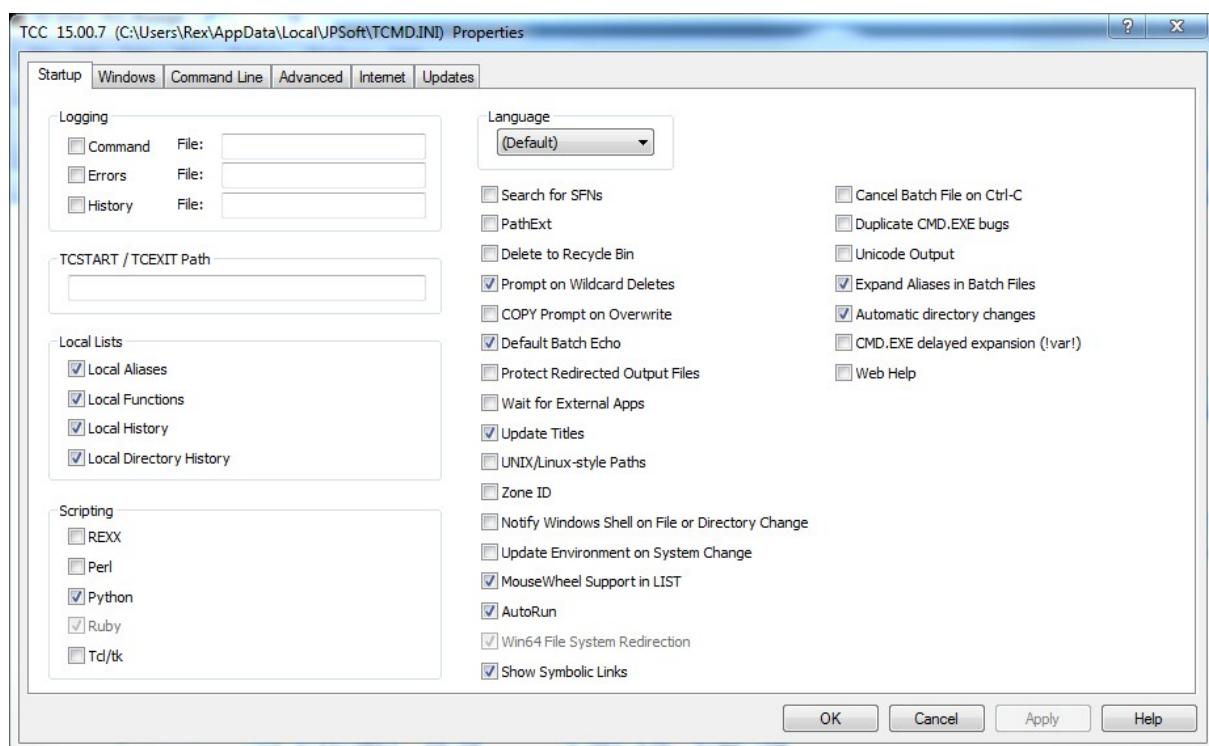
For details about the *.INI* file and *.INI* file directives, the allowable ranges for each, and the effect of each, see [Initialization \(.INI\) Files](#) and [Directives](#).

While you are using the dialog, you can move between sets of configuration options by clicking on the individual tabs. The options available in this dialog are:

[Startup](#)  
[Windows](#)  
[Command Line](#)  
[Advanced](#)  
[Internet](#)  
[Updates](#)

#### 4.9.2.1 Startup

If you are not familiar with the purpose or use of the Startup configuration dialog, review the main [configuration dialogs](#) topic before continuing.



#### Logging:

**Command** : Save internal and external commands (after alias and variable expansion) executed either from the command prompt or a batch file to the log file. See [LOG](#) for more details.

**Errors** : Save error messages to the log file. If you enter a file name in the File field, that file will be used for error logging. See [LOG](#) for more details.

**History** : Save each command executed from the command prompt exactly as it was entered (before aliases and variable expansion) to the log file. If you enter a file name in the File field, that file will be used for history logging. See [LOG](#) for more details.

#### TCSTART / TCEXIT:

You can set the path to your [TCSTART / TCEXIT](#) files if they aren't in the same directory as **Take**

**Command.****Local Lists:**

**Local Aliases** instructs **TCC** to use a local or (if unchecked) global alias list.

**Local Functions** instructs **TCC** to use a local or (if unchecked) global function list.

**Local History** instructs **TCC** to use a local or (if unchecked) global command history list.

**Local Directory History** instructs **TCC** to use a local or (if unchecked) global directory history.

**Scripting:****Not in LE**

**REXX** : Enable the internal [REXX](#) support (Open Object REXX).

**Perl** : Enable the internal [Perl](#) (ActiveState 5.10) support.

**Python** : Enable the internal [Python](#) (ActivePython 2.6) support.

**Ruby** : Enable the internal [Ruby](#) (1.8 or 1.9) support.

**Tcl** : Enable the internal [Tcl](#) (8.5.7) support.

Note : you must restart the **TCC** tab window for the **REXX**, **Perl**, **Python**, **Ruby**, and **Tcl** options to take effect.

**Language:**

The **Language** combo box allows you to override the default language that **Take Command** uses for menus and dialogs.

**Search for SFNs** : If enabled, filename searches will search for both long filenames and short filenames. See [LFN File Searches](#) for details.

**PathExt** : Determines whether **TCC** will use the PATHEXT environment variable. If disabled, the PATHEXT variable is ignored. If enabled, the [PATHEXT](#) variable will be used to determine extensions to look for when searching the PATH for an executable file. For details, see the [PATHEXT](#) variable and the [PATH](#) command. If you enable **PathExt** and then fail to set the PATHEXT variable, path searches will fail as there will be no extensions for which to search!

**Delete to Recycle Bin** : If enabled, files deleted by the [DEL / ERASE](#) commands and by [RD /S](#) are placed in the Windows Recycle Bin. If disabled, the files are deleted without being placed in the Recycle Bin. [DEL](#)'s and [RD](#)'s /K and /R switches allow you to override this setting for individual commands. The [RecycleExclude](#) environment variable can be used to exclude specific files.

**Prompt on Wildcard Deletes** : Enable the confirmation prompt from [DEL /Q](#) when doing a wildcard-only or directory deletion. Use caution if you disable this option, as this will allow DEL /Q to delete an entire directory without prompting for confirmation. See [DEL](#) for additional details.

**Copy Prompt on Overwrite** : If enabled [COPY](#) and [MOVE](#) will prompt before overwriting an existing file if the command is being performed at the command prompt. (This duplicates the behavior of the current version of CMD.)

**Default Batch Echo** : Set the default batch echo mode. If enabled, all batch file commands are echoed unless [ECHO](#) is explicitly set off in the batch file. If disabled, no batch file commands are echoed unless [ECHO](#) is explicitly set on. See also: [SETDOS /V](#).

**Protect Redirected Output Files** : If enabled, standard output [redirection](#) will be prevented from overwriting an existing file, and will require that the output file already exist for append redirection. (You can override this option by adding the exclamation point to the output redirection symbol; i.e. `>!`.) See also: [SETDOS /N](#).

**Wait for External Apps** : Determines whether **TCC** waits for an external program started from the command line to complete before redisplaying the prompt. See [Waiting for Applications to Finish](#) for details on the effects of this option.

**Update Titles** : *Take Command* normally changes the window titles to include the command or batch file name each time a new command is executed. If you prefer a static title bar which does not change with each command, disable this option.

**UNIX/Linux-style Paths** : Enables the forward slash as a path separator in the command name (the first item on the command line). Note that setting UnixPaths to Yes does not change the switch character, it simply allows you to put forward slashes in the command name. When this option is enabled, command switches beginning with a forward slash must be preceded by a space to avoid confusion (this is a good general practice).

**Zone ID** : Set the NTFS Zone ID security when running executables downloaded from the Internet. (Note that CMD never checks for the Zone ID, so setting it may introduce a minor incompatibility.)

**Notify Windows Shell on File or Directory Change** : Notify the system shell when changing files or directories. The shell notification is done by the [ASSOC](#), [COPY](#), [DEL](#), [MD](#), [MOVE](#), and [RD](#) commands. Note that setting this option could introduce a slight incompatibility with CMD, which doesn't notify the system shell about anything.

**Update Environment on System Change** : If enabled, **TCC** will monitor the WM\_SETTINGCHANGE message and if the environment is specified, update the environment from the User, Volatile, and System registry entries. The updates are done whenever **TCC** displays a prompt (to prevent the environment from changing in the middle of a batch file). Unless you have a specific need for this option it's better not to enable it, as it can result in variables set by **TCC**'s parent process being destroyed.

**MouseWheel Support in LIST** : Set mouse wheel support in [LIST](#). Disable this option if you experience incompatibilities with other applications.

**AutoRun** : If enabled when a **TCC** tabbed window starts, execute the AutoRun registry variables (HKEY\_LOCAL\_MACHINE\Software\Microsoft\Command Processor\AutoRun and/or HKEY\_CURRENT\_USER\Software\Microsoft\Command Processor\AutoRun).

**Win64 File System Redirection** : If disabled, overrides the default Win64 behavior of remapping windows\system32 calls to windows\SysWOW64.

**Show Symbolic Links** : Displays the symbolic link in [DIR](#) or [PDIR](#) (Windows Vista or later only).

**Cancel Batch File on Ctrl-C** : Cancel batch file processing without the usual prompt when you press Control-C.

**Duplicate CMD bugs** : Tells the **TCC** parser to duplicate bugs in CMD. The only bugs currently replicated are in the [FOR](#) and [IF](#) commands.

**Unicode Output** : The **TCC** output files (such as redirected output) will be written in Unicode format.

**Expand Aliases in Batch Files** : If disabled, **TCC** won't try to expand command aliases when in a batch file. (Directory aliases will still be expanded.)

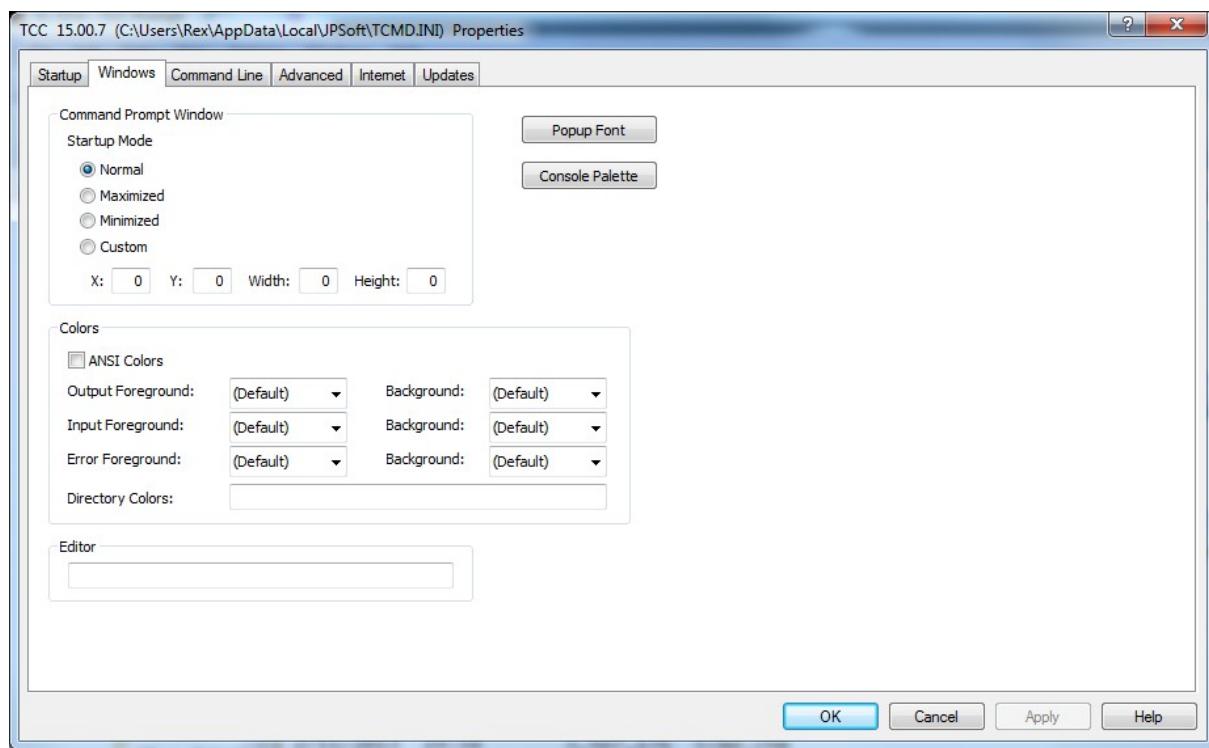
**Automatic Directory Changes** : If enabled, **TCC** will change directories when a directory name with a trailing \ is the only argument on the command line.

**CMD delayed variable expansion (!var!)** : If enabled, **TCC** will emulate the peculiar CMD !var! expansion.

**Web Help** : If enabled, **TCC** will use the browser-based help (at <http://jpsoft.com/help/index.htm>) instead of the local help. Using web help allows you to add comments to the help topics.

#### 4.9.2.2 Windows

If you are not familiar with the purpose or use of the Windows configuration dialog, review the main [configuration dialogs](#) topic before continuing.



#### Command Prompt Window

(These options are only used when **TCC** is not running in a *Take Command* tab window.)

The **Normal**, **Max**, **Min**, and **Custom** buttons select the initial state for the **TCC** window.

The **X**, **Y**, **Width**, and **Height** fields set the initial size and position of the **TCC** window. They are

ignored unless the **Custom** button is also selected.

#### Colors:

**ANSI Colors** : Enable ANSI X3.64 string processing of the output of Take Command internal commands. Note that ANSI X3.64 processing of the output of external applications is not supported. See the [ANSI X3.64 Commands Reference](#) for a list of the ANSI X3.64 commands supported by **TCC**.

**Colors** : Select foreground and background colors for input, output, and error messages.

**Directory Colors** : Sets the directory colors used by [DIR](#) and [SELECT](#). The format is the same as that used for the [COLORDIR](#) environment variable. See [Color-Coded Directories](#) for a detailed explanation.

#### Editor:

**Editor** : The pathname of the editing program to run from [LIST](#) if there is no Windows "edit" association for the extension. (The default is NOTEPAD.EXE.)

#### Pop-Up Font:

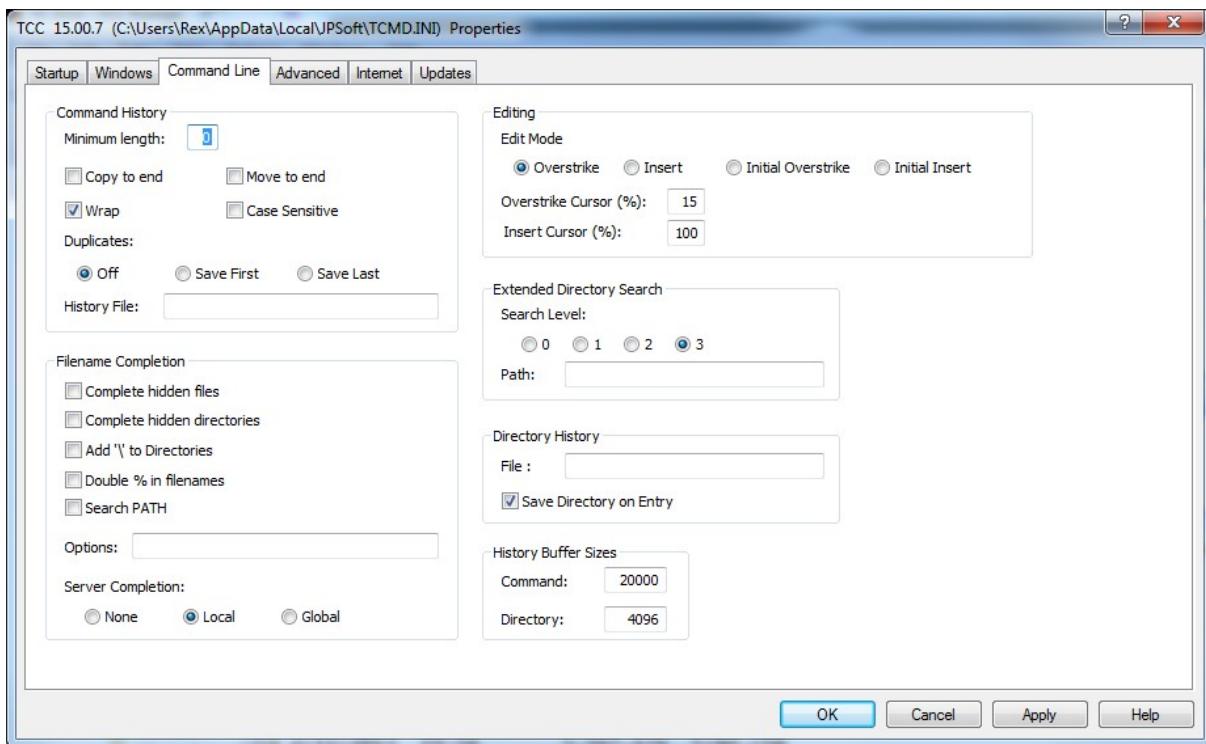
Set the font to use in the command history, directory history, filename completion, and fuzzy directory searching popup windows.

#### Console Palette:

Defines a custom color palette, not restricted to the standard console window 16 colors. (Requires Windows Vista or later.) This will **not** work when running **TCC** in a **Take Command** tab window due to a Windows API bug. (But you can define custom palettes in **Take Command** for tab windows.)

### 4.9.2.3 Command Line

If you are not familiar with the purpose or use of the Command Line configuration dialog, review the main [configuration dialogs](#) topic before continuing.



### Command History:

**Minimum Length :** Set the minimum command line size to save in the command history list. Any command line whose length is less than this value will not be saved. Legal values range from 0, which saves everything, to 8192. You can prevent any command line from being saved in the history by beginning it with an "at" sign ("@"). See also the [HISTORY](#) command.

**Copy to end :** This option controls what happens when you re-execute a line from the command history. If this option is set, the line is appended to the end of the history list. The original copy of the command is always retained at its original position in the list. If this option is set, it will override **Move to End**.

**Move to end :** If enabled, a recalled line will be moved to the end of the command history. The difference between this directive and **Copy to end** is that **Copy to end** copies each recalled line to the end of the history but leaves the original in place. **Move to end** copies the line to the end of history and removes the original line. This directive has no effect if **Copy to end** is set.

**Wrap :** If enabled, the command history "wraps" when you reach the top or bottom of the list (so the list appears "circular"). If this option is disabled, history recall will stop (and beep) at the beginning and end of the list rather than wrapping.

**Case Sensitive :** If enabled, the command history comparisons will be case sensitive.

**Duplicates :** Controls duplicate entry placement in the [history list](#).

**Off**

Always add new entries

**Save First**

Add new entry only if it does not match any old entries

**Save Last**

Add new entry unconditionally, and delete matching older entries

**History File** : Load the specified history list file before running [TCSTART](#), and save the command history to the file after running [TCEXIT](#). You should include the full pathname.

#### Filename Completion:

**Complete hidden files** : If enabled, hidden and system files will be displayed by [filename completion](#).

**Complete hidden directories** : If enabled, hidden directories will be displayed by [filename completion](#). [CDD /S](#) will also index hidden directories if this option is set.

**Add '\' to Directories** : If enabled, a \ (backslash) is automatically appended to directory names (or / to FTP directories) in [filename completion](#).

**Double % in filename** : If enabled, and the filename has embedded % characters, and the first argument on the command line is an internal command, the % characters will be doubled so that variable expansion won't delete (or unexpectedly expand) the filename. (This will not affect filenames on lines beginning with aliases or variables.)

**Search PATH** : If enabled, the directories in the [PATH](#) variable are searched if a match isn't found in the current directory.

**Options** : Sets the files returned during [filename completion](#) for selected commands. The format is the same as that used for the [FILECOMPLETION](#) environment variable. Note that specifying any options for a command here will override the default filename completion options (such as complete hidden files & directories). See [Customizing Filename Completion](#) for a detailed explanation of selective filename completion.

**Server Completion** : Configures server name completion (see [Filename Completion](#) for information on how to use server name completion). **Local** lists only local servers (i.e., those in your "network neighborhood"). **Global** will enumerate the entire network. **None** will disable server completion; this may be necessary to prevent "hanging" if you start typing a server name and accidentally press Tab, and your local domain is very large or slow to respond.

#### Editing:

**Edit Mode** : Starts the command line editor in either **Insert** or **Overstrike** mode. If you specify **Initial Overstrike** or **Initial Insert**, the command line editor will start in the specified state, but if you toggle insert mode while editing a line, the editor will continue to use the new mode on subsequent lines. See also: [SETDOS /M](#).

**Overstrike Cursor** : The shape of the cursor for insert mode during command line editing, and all commands which accept line input (DESCRIBE, ESET, etc.). The size is a percentage of the total character cell size, between 0% and 100%. Because of the way video drivers map the cursor shape, you may not get a smooth progression in cursor shapes as **Insert Cursor** and **Overstrike Cursor** change. If you set **Insert Cursor** and **Overstrike Cursor** to -1, the cursor shape won't be modified at all. If you set them to 0, the cursor will be invisible. See also: [SETDOS /S](#).

**Insert Cursor** : The shape of the cursor for overstrike mode during command line editing and all commands which accept line input. The size is a percentage of the total character cell size, between 0% and 100%. See also: **Overstrike Cursor** (above) and [SETDOS /S](#).

**Extended Directory Search:**

**Search Level :** Configure extended directory searches. **0** disables extended searches. For complete details on the meaning of the other settings see [Extended Directory Searches](#).

**Path :** The path to *JPSTREE.IDX*, the file used for the [extended directory search](#) database.

**Directory History :**

**File :** Load the specified directory history list file before running [TCSTART](#), and save the directory history to the file after running [TCEXIT](#).

**Save Directory on Entry:** *TCC* normally saves the previous directory to the directory history when you change to a new directory. This option saves the new directory to the directory history when you change directories.

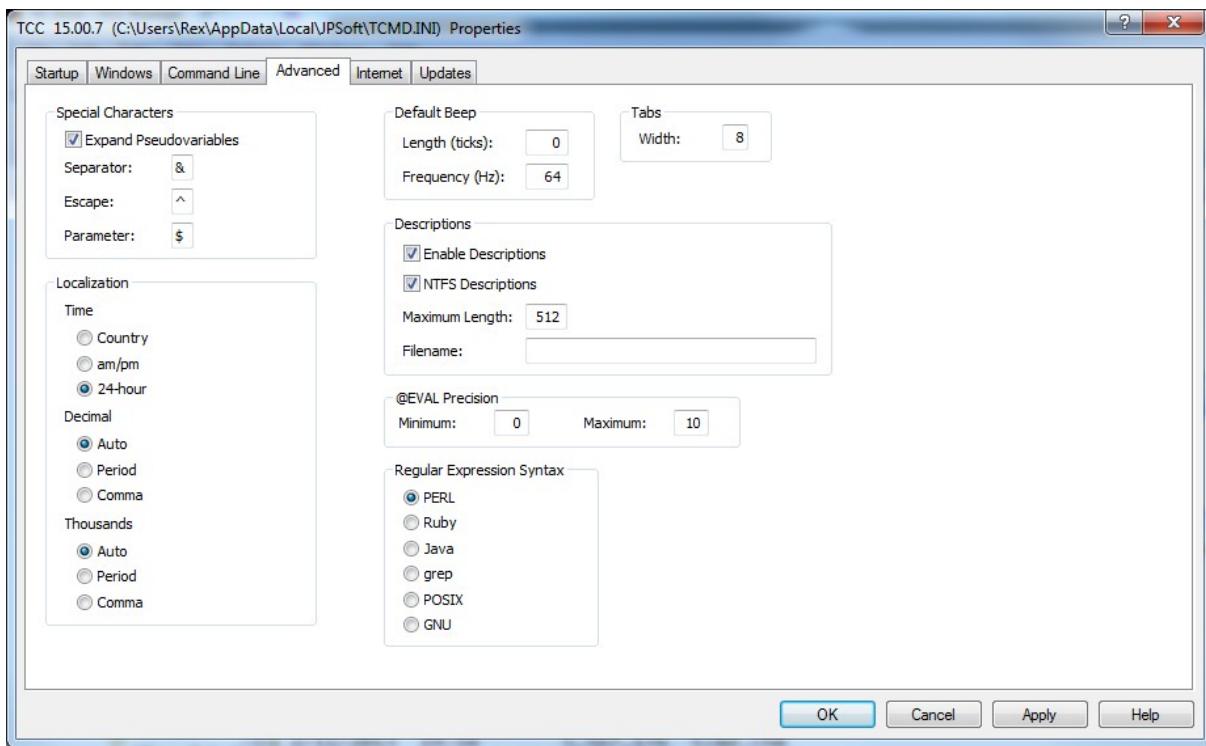
**History Buffer Sizes**

**Command History :** Set the amount of memory allocated to the [command history](#) list (in characters). The allowable range of values is from 4,000 to 500,000. If you use a [global history list](#), this value is ignored in all sessions except that which first establishes the global list. (To change the size, you will need to close all of the *TCC* windows, and any [SHRALIAS](#) session.)

**Directory History :** Set the amount of memory allocated to the [directory history](#) list (in characters). The allowable range is 1,000 to 50,000. If you use a [global directory history](#) list, this value is ignored in all sessions except that which first establishes the global list. (To change the size, you will need to close all of the *TCC* windows, and [SHRALIAS](#) session.)

#### 4.9.2.4 Advanced

If you are not familiar with the purpose or use of the Advanced configuration dialog, review the main [configuration dialogs](#) topic before continuing.



### Special Characters:

**Not in LE**

**Expand Pseudovariables:** If set, TCC will expand the [%\\_](#) and [%+](#) pseudovariables. (You can disable this option if you're looking for maximum CMD compatibility.)

**Separator :** The character used to separate multiple commands on the same line. The default is an ampersand (**&**). It can be dynamically modified by the /C option of the [SETDOS](#) command. You cannot use any of the [redirection](#) characters (| > <) or any of the white space characters (space, tab, comma, or equal sign).

**Escape :** The character used to suppress the normal meaning of the following character. The default is a caret (^). See [Escape Character](#) for a description of the special escape sequences. You cannot use any of the [redirection](#) characters (|, >, or <) or the white space characters (space, tab, comma, or equal sign) as the escape character. See also: [SETDOS](#) /E.

**Parameter :** The character used after a percent sign to specify all or all remaining command line parameters in a batch file or alias (e.g., %\$ or %n\$; see [Batch File Parameters](#) and [ALIAS](#)). The default is the dollar sign [**\$**]. See also: [SETDOS](#) /P.

### Localization:

**Time :** The format of time displays in the output of the [DATE](#), [DIR](#), [SELECT](#), [TIME](#) and [TIMER](#) commands, and in [LOG](#) files. It has no effect on [%\\_ TIME](#), [%@MAKETIME](#), the [\\$t](#) and [\\$T](#) options of [PROMPT](#), or date and time [ranges](#).

|                |                                                                               |
|----------------|-------------------------------------------------------------------------------|
| <b>Country</b> | Formats the time according to the country code set for your system.           |
| <b>am/pm</b>   | Displays the time in 12-hour format with a trailing "a" for AM or "p" for PM. |
| <b>24-hour</b> | Display <b>the time</b> in 24-hour time format.                               |

**Decimal** : Sets the character used as the decimal separator for [@EVAL](#), numeric [IF](#) and [IFF](#) tests, version numbers, and other similar uses. The only valid settings are period [.], comma [,], and **Auto** (the default). A setting of **Auto** tells **TCC** to use the decimal separator associated with your current country code. If you change the decimal character you will need to adjust the thousands character so that the two characters are different. See also: [SETDOS /G](#).

**Thousands** : Sets the character used as the thousands separator for numeric output. The only valid settings are period [.], comma [,], and **Auto** (the default). **Auto** tells **TCC** to use the thousands separator associated with your current country code. If you change the thousands character you will need to adjust the decimal character so that the two characters are different. See also: [SETDOS /G](#).

#### Default Beep:

**Length** : The default [BEEP](#) length in system clock ticks (approximately 1/18 of a second per tick). Also the default length for "error" beeps (for example, if you press an illegal key).

**Frequency** : The default frequency (in Hz) for the [BEEP](#) command. This is also the frequency for "error" beeps (for example, if you press an illegal key). To disable all error beeps set the length and frequency to 0. If you do, the [BEEP](#) command will still be operable, but will not produce sound unless you explicitly specify the frequency and duration.

You can play a system sound on an error by setting Length to 0 and Frequency to the desired sound:

- 0** Windows default ("OK") beep sound
- 16** Windows Critical Stop ("Hand") sound
- 32** Windows Question sound
- 48** Windows Exclamation sound
- 64** Windows Asterisk sound

#### Tabs:

**Tabs** : Sets the tab stops for [LIST](#) output. The allowable range is 1 to 32.

#### Descriptions:

**Enable Descriptions** : Set description handling for the file processing commands (COPY, DEL, MOVE, REN, etc.). If disabled, **TCC** will not update the [description](#) file when files are moved, copied, deleted or renamed. See also: [SETDOS /D](#).

**NTFS Descriptions** : If set, **TCC** uses the Comments field in the NTFS SummaryInformation stream for each file to hold its description, instead of the [DESCRIPT.ION](#) file. The advantages are that the description will always remain with the file regardless of what program copies, moves, or renames it. The disadvantage is that you cannot attach a description to directories.

**Maximum Length** : Set the description length limit for [DESCRIBE](#). The allowable range is 20 to 511 characters.

**Filename** : Sets the file name in which to store file descriptions. The default file name is [DESCRIPT.ION](#). See also: [SETDOS /D](#).

#### @EVAL Precision

**Minimum** : The minimum number of digits after the decimal point in values displayed by [@EVAL](#).

The allowable range is 0 to 1000. This directive will be ignored if **Minimum** is larger than

**Maximum**. You can override this setting with the construct `@EVAL[expression=n,n]`. See also: [SETDOS /F](#).

**Maximum** : The maximum number of digits after the decimal point in values displayed by [@EVAL](#).

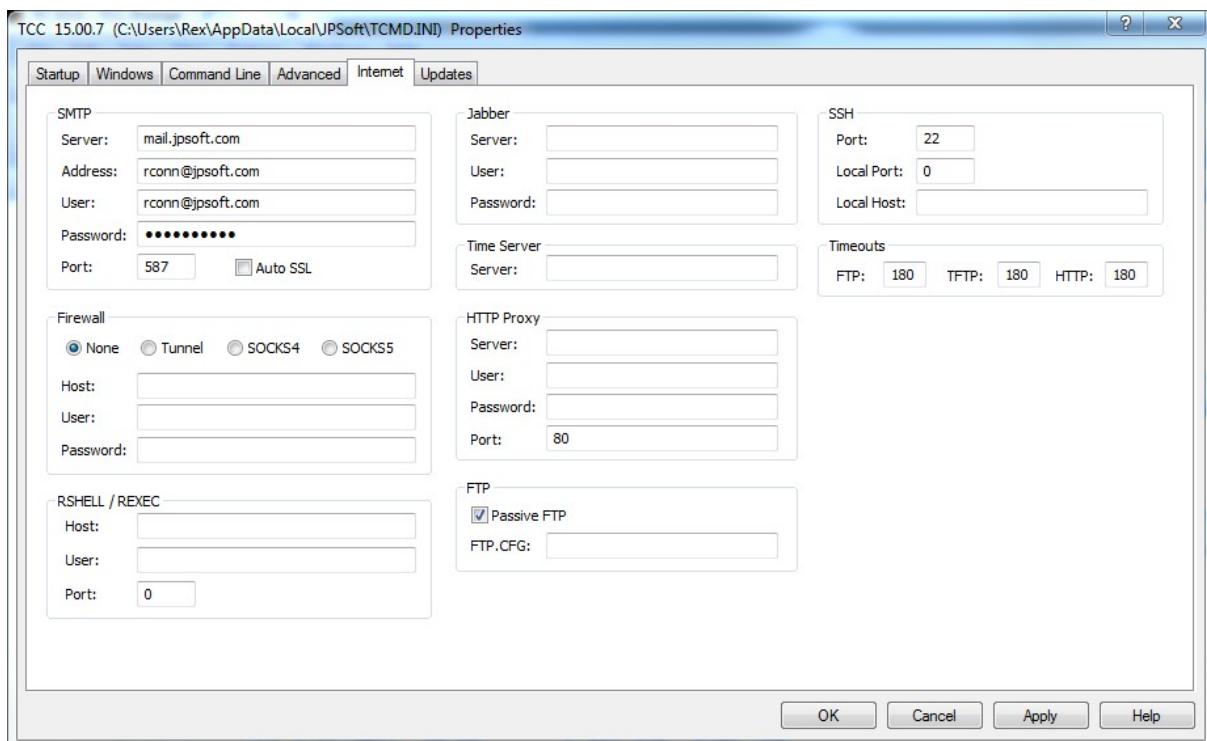
You can override this setting with the construct `@EVAL[expression=n,n]`. The allowable range is 0 to 1000; if you use the "`=n,n`" syntax the maximum is 10,000. See also: [SETDOS /F](#).

## Regular Expression Syntax

Sets the type of [regular expression](#) syntax to use.

### 4.9.2.5 Internet

If you are not familiar with the purpose or use of the Internet configuration dialog, review the main [configuration dialogs](#) topic before continuing.



#### SMTP:

**Server** : The SMTP server name to use in [SENDMAIL](#) and [SENDHTML](#) for outgoing mail. (If not set, **TCC** will attempt to get the address from the registry.)

**Address** : The email address to use in [SENDMAIL](#) and [SENDHTML](#) for outgoing mail. (If not set, **TCC** will attempt to get the current user's email address from the registry.)

**User** : The email username (if your SMTP server requires it for authentication).

**Password** : The email password of the user (if your SMTP server requires it for authentication).

**Port** : The SMTP port number for use by [SENDMAIL](#) and [SENDHTML](#). (The default port number is 25).

**Auto SSL** : If checked, [SENDMAIL](#) and [SENDHTML](#) will use Auto SSL negotiation. If the remote port is set to the standard plaintext port, **TCC** will use Explicit mode. In all other cases, SSL negotiation will be implicit.

#### Firewall:

**Type** : The type of firewall in use on your network.

**Host** : The server name of the firewall (if any) for FTP and HTTP access.

**User** : The user name if the firewall requires authentication.

**Password** : The password if the firewall requires authentication.

#### RSHELL / REXEC:

**Host** : The name of the local host or user-assigned IP interface through which connections are initiated or accepted (for [REXEC](#) and [RSHELL](#)).

**User** : The name of the user on the local machine (for [RSHELL](#)).

**Port** : The port number (or communication endpoint) in the local machine to bind to for [REXEC](#) and [RSHELL](#).

#### Timeouts:

Set the timeout (inactivity) period in seconds for FTP, TFTP, and HTTP operations.

#### JABBER:

**Server** is the [JABBER](#) server to log into.

**User** : The default user name for logging onto a Jabber server and sending IM's via the [JABBER](#) command.

**Password** : The logon password for the default Jabber user.

#### Time Server:

**Time Server** : The URL for the internet time server for [TIME /S](#). If no server is specified, TIME uses clock.psu.edu.

#### HTTP Proxy:

**Server** : The proxy server address to use for HTTP calls

**User** : The user name if Basic authentication is to be used for the HTTP proxy.

**Password** : The user password if Basic authentication is to be used for the HTTP proxy.

**Port :** The proxy port number to use for HTTP calls.

**FTP:**

**Passive FTP :** Set passive mode for FTP calls (sometimes required by a firewall).

**FTP.CFG :** Specify the location and name of the file containing the FTP user names and passwords, and optionally the directory format for non-standard FTP servers. The default is FTP.CFG in the *Take Command* installation directory. See [Using FTP/HTTP Servers](#) for details.

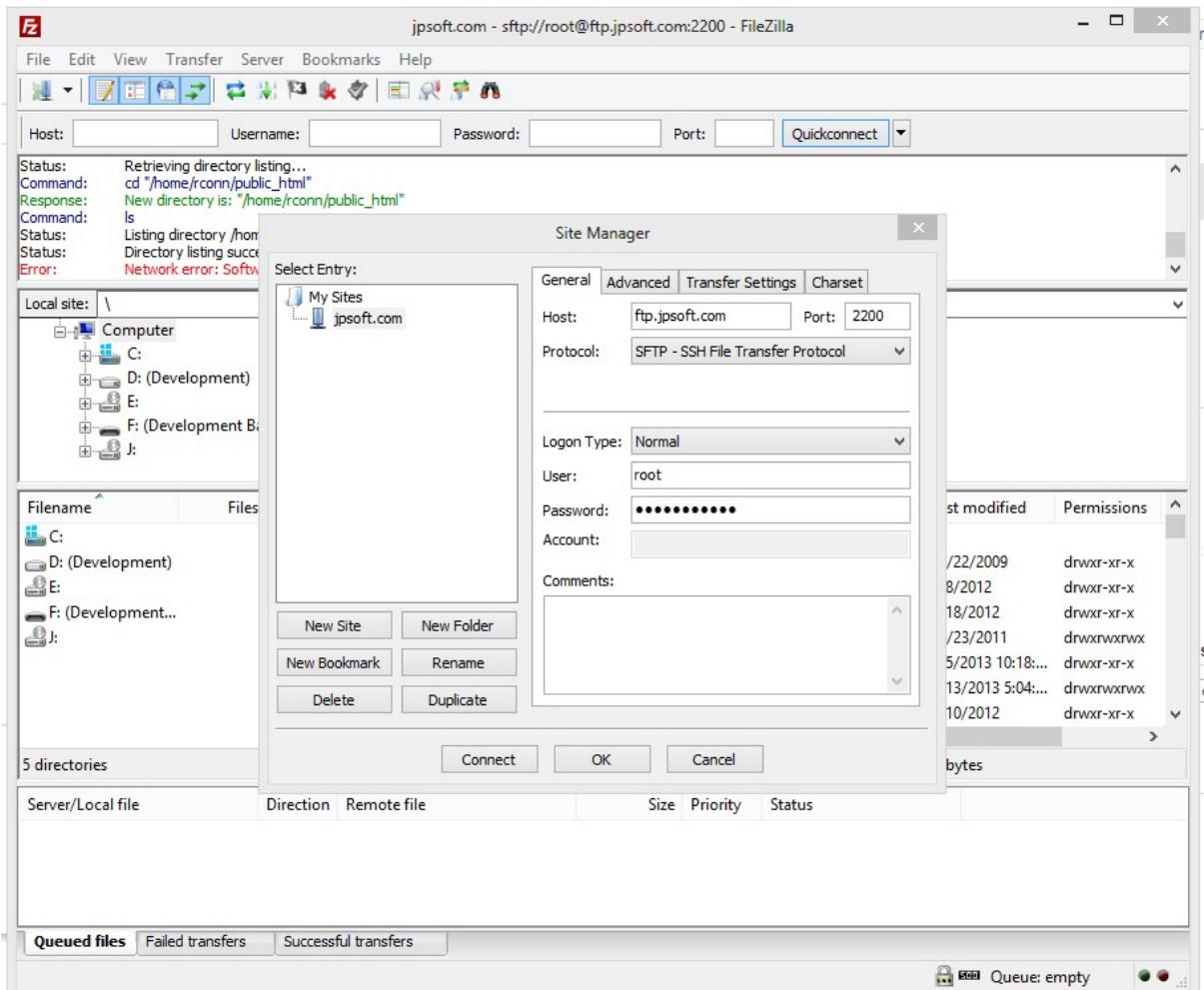
**SSH:**

**SSHPort :** The port on the SSH server where the SSH service is running (default is 22).

**SSHLocalPort :** The TCP port in the local host where IPPort binds.

**SSHLocalHost :** The name of the local host or user-assigned IP interface through which connections are initiated or accepted.

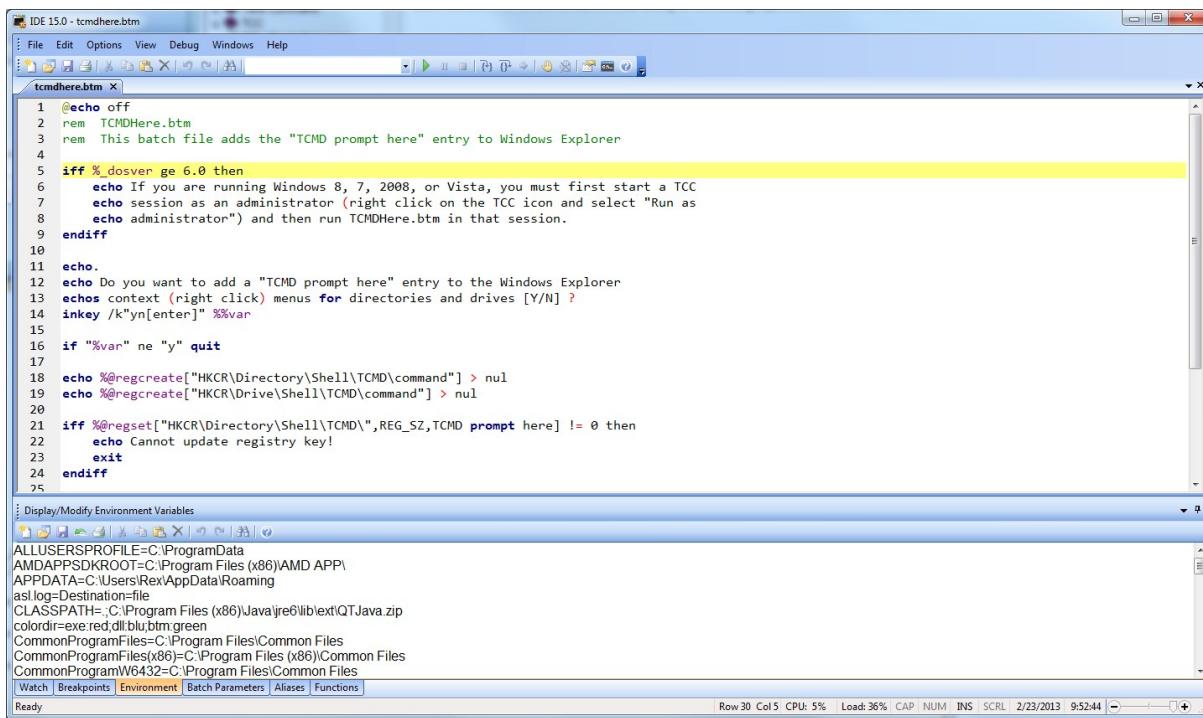
#### 4.9.2.6 Updates



The **Check for Updates** button queries the JP Software web server to see if there is an updated version of **Take Command / TCC** available. If there is, the new version information will be displayed and you can choose to download and automatically update your existing version.

## 5 IDE / Batch Debugger

**Take Command** has a very powerful IDE (Integrated Development Environment) for creating, editing, and debugging batch files. The IDE includes syntax coloring for batch files (.BAT, .BTM, and .CMD) and code folding for command groups, DO, IFF, SWITCH, and TEXT.



The IDE window includes a slider control on the lower right corner of the status bar to control the transparency level.

If a file in a tab window has been modified but not yet saved, the tab title will be prefixed with a \*. When the file is saved, the \* is removed.

The edit window toolbar (which is configurable by clicking on the rightmost down arrow), has a number of icons to control debugging. Each has a tooltip for quick reference:

|                          |                                                                                                                          |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>New</b>               | Create a new batch file in a new tab window.                                                                             |
| <b>Open</b>              | Open an existing batch file in a new tab window.                                                                         |
| <b>Save</b>              | Save the current batch file.                                                                                             |
| <b>Print</b>             | Print the current batch file.                                                                                            |
| <b>Cut</b>               | Copy the highlighted selection to the clipboard and delete it from the file.                                             |
| <b>Copy</b>              | Copy the highlighted selection to the clipboard.                                                                         |
| <b>Paste</b>             | Copy the contents of the clipboard to the current cursor location.                                                       |
| <b>Delete</b>            | Delete the highlighted selection.                                                                                        |
| <b>Undo</b>              | Undo the last edit.                                                                                                      |
| <b>Redo</b>              | Restore the last Undo.                                                                                                   |
| <b>Find</b>              | Search for text.                                                                                                         |
| <b>Batch Arguments</b>   | New batch file arguments. The text will be parsed into %1 - %n batch arguments and used when the batch file is debugged. |
| <b>Start Debugging</b>   | Starts the debugger. The cursor will be placed on the first line.                                                        |
| <b>Pause Debugging</b>   | Pause execution at the next line.                                                                                        |
| <b>Stop Debugging</b>    | Stops the debugger.                                                                                                      |
| <b>Step Into</b>         | Execute the current line.                                                                                                |
| <b>Step Over</b>         | Execute the current line but disable the debugger during a CALL or GOSUB.                                                |
| <b>Run to Breakpoint</b> | Execute the batch file, stopping at the next breakpoint.                                                                 |
| <b>Toggle Breakpoint</b> | Sets or turns off a breakpoint on the current line.                                                                      |

|                          |                                                                                                             |
|--------------------------|-------------------------------------------------------------------------------------------------------------|
| <b>Clear Breakpoints</b> | Clear all breakpoints in the current batch file.                                                            |
| <b>File Properties</b>   | Displays information on the current batch file.                                                             |
| <b>Start New Shell</b>   | Start another copy of <b>TCC</b> (this is useful if you need to perform some tasks while debugging a file.) |
| <b>Help</b>              | Display the online help.                                                                                    |

The IDE includes tabbed windows to display and/or modify the watch list, aliases, batch variables, environment variables, and user functions. The variable windows also have a toolbar, with the following buttons:

|               |                                                                                            |
|---------------|--------------------------------------------------------------------------------------------|
| <b>New</b>    | Restore the original values for the list (alias, variable, environment, or function)       |
| <b>Open</b>   | Add the contents of a file to the list                                                     |
| <b>Save</b>   | Save the current list to a file                                                            |
| <b>Apply</b>  | Replace the original values with the modified list                                         |
| <b>Print</b>  | Print the current list                                                                     |
| <b>Cut</b>    | Copy the highlighted selection to the clipboard and delete it from the file                |
| <b>Copy</b>   | Copy the highlighted selection to the clipboard                                            |
| <b>Paste</b>  | Copy the contents of the clipboard to the current cursor location                          |
| <b>Delete</b> | Delete the highlighted selection (or the character at the cursor location if no selection) |
| <b>Undo</b>   | Undo the last edit                                                                         |
| <b>Redo</b>   | Restore the last Undo                                                                      |
| <b>Find</b>   | Search for text                                                                            |
| <b>Help</b>   | Display the online help                                                                    |

The environment variables window displays any variables changed since the last command in red. You can specify variables to exclude from the environment variable window with the `DebugVariableExclude` variable. For example, to suppress the display of the processor and user variables:

```
set DebugVariableExclude=proc*;user*
```

Note that this option doesn't affect the existence of the variables, just whether they're displayed in the environment variable window.

If you right click on the first column in the Watch window, the debugger will display an environment variable listbox. Select an entry to have it added to the watch list.

If you press **Ctrl-C** or **Ctrl-Break** while debugging, you will see the prompt:

```
Cancel batch job filename (Y/N/A/D) :
```

Pressing **D** will return you to single-step mode in the debugger. (This allows you to interrupt a **run-to-breakpoint** without terminating the debugger and batch file.)

## Margins

There are three possible margins on the left of the edit window:

- The line number (selectable by the "Options / Display Line Numbers" menu option).
- The Breakpoint margin (left click in this margin to set a breakpoint on this line).
- The Fold margin (selectable by the "Options / Display Fold Margin" menu option), which will display a - for blocks that can be collapsed to a single line (DO, IFF, and SWITCH commands, and command groups). When a block is collapsed, the Fold margin will display a +. Left clicking

in the Fold margin will toggle the fold state.

## Edit Windows

The text processing commands available in the IDE edit windows are listed below. The text commands can be classified into general categories:

- [Caret commands](#)
- [Edit commands](#)
- [Mark / Clipboard commands](#)
- [Search commands](#)
- [File commands](#)
- [Bookmark commands](#)
- [Breakpoint commands](#)
- [Expression evaluation commands](#)

### • Caret commands

|                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Right</b>            | This command will move the caret one character to the right. When the caret is on the last position of the current line it is moved to the first position of the next line.                                                                                                                                                                                                                                                                                                                               |
| <b>Shift-Right</b>      | In addition to the caret movement this command will also extend the current selection to the new caret position.                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Left</b>             | This command will move the caret one character to the left. When the caret is on the first position of the current line it is moved to the last position of the previous line.                                                                                                                                                                                                                                                                                                                            |
| <b>Shift-Left</b>       | In addition to the caret movement, this will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Up</b>               | This command will move the caret one line up. The caret column position will be set as close to its previous column position as possible.                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Shift-Up</b>         | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Down</b>             | This command will move the caret one line down. The caret column position will be set as close to its previous column position as possible. When the caret is on the last line but not on the last column it will be moved to the last column.                                                                                                                                                                                                                                                            |
| <b>Shift-Down</b>       | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>End</b>              | This command will move the caret to the end of the line it is currently on. If the caret is already at the end nothing happens.                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Shift-End</b>        | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Home</b>             | This command will move the caret to the start of the line it is currently on. If the caret is already at the start nothing happens.                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Shift-Home</b>       | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Ctrl-Right</b>       | This command will move in one of the following ways: <ul style="list-style-type: none"><li>• When the caret is located on a delimiter character the caret is moved right until the first non-delimiter is found.</li><li>• When the caret is located on a non-delimiter character the caret is moved to the next delimiter character.</li><li>• When the caret is located on the last word or delimiter of the current line the caret is moved to the first word or delimiter of the next line.</li></ul> |
| <b>Ctrl-Shift-Right</b> | In addition to the caret movement this command will also extend the current selection to the new caret position.                                                                                                                                                                                                                                                                                                                                                                                          |

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Ctrl-Left</b>       | This command will move in one of the following ways: <ul style="list-style-type: none"> <li>• When the caret is located on a delimiter character the caret is moved to the start of the previous word.</li> <li>• When the caret is located on a non-delimiter character and not on a white-space character the caret is moved to the start of the current word.</li> <li>• When the caret is located on the start of the first word, delimiters or white-space of the current line the caret is moved to the start of the last word or delimiters of the previous line.</li> </ul> |
| <b>Ctrl-Shift-Left</b> | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Ctrl-Home</b>       | This command will move the caret to the beginning of the text. When the caret is already at this location nothing happens.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Ctrl-Shift-Home</b> | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Ctrl-End</b>        | This command will move the caret to the end of the text. When the caret is already at this location nothing happens.                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Ctrl-Shift-End</b>  | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>PgUp</b>            | This command will move the caret one view up when it is located on the top line currently in the view. When the caret is not located on the top line of the view, it will be moved there.                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Shift-PgUp</b>      | In addition to the caret movement this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>PgDn</b>            | This command will move the caret one view down when it is located on the bottom line currently in the view. When the caret is not located on the bottom line of the view, it will be moved there.                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Shift-PgDn</b>      | In addition to the caret movement, this command will also extend the current selection to the new position.                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

- **Edit commands**

|                       |                                                                                                                                                                                                                                                                                                                    |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Ctrl-Z</b>         | This command will undo the last change made to the edit control contents. You can undo any number of changes made to the control contents up to the maximum number of undo/redo hops.                                                                                                                              |
| <b>Ctrl-Y</b>         | This command will redo the last change you have undone. You can re-do any number of changes up to the number of changes undone.                                                                                                                                                                                    |
| <b>Backspace</b>      | This command will remove the character to the left of the caret. When the caret is located at the start of the line, the characters right of the caret are appended to the previous line and the caret is moved to be positioned between the old line contents and the appended characters.                        |
| <b>Delete</b>         | This command removes the character to the right of the caret. When there are no characters to the right of the caret, the contents of the next line is appended to the current line.                                                                                                                               |
| <b>Return</b>         | This command will split the current line and create a new line of the characters, if any, right of the caret. The caret is moved to the start of the newly created line.                                                                                                                                           |
| <b>Ctrl-Delete</b>    | When the caret is located on a word, this command will delete all characters in the word right of the caret position.                                                                                                                                                                                              |
| <b>Ctrl-Backspace</b> | When the caret is located on a word, this command will delete all characters in the word left of the caret position.                                                                                                                                                                                               |
| <b>Tab</b>            | This command does one of the two following things: <ul style="list-style-type: none"> <li>• When there is a valid text selection, this command will indent the lines covered by the selection right by one tab-stop.</li> <li>• When there is no text selection, a tab is inserted at the current caret</li> </ul> |

|                     |                                                                                                                                                                                  |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Shift-Tab</b>    | position.<br>When there is a valid text selection, this command will indent the lines covered by the selection left by one tab-stop.                                             |
| <b>Shift-Ctrl-U</b> | When there is a valid selection, this command will convert all lower-case characters in the selection to upper-case characters. If there is no valid selection, nothing happens. |
| <b>Ctrl-U</b>       | When there is a valid selection, this command will convert all upper-case characters in the selection to lower-case characters. If there is no valid selection, nothing happens. |
| <b>Ins</b>          | This command will toggle the current editing mode between overwrite and insert.                                                                                                  |

- **Mark / Clipboard commands**

|               |                                                                                                                               |
|---------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>Ctrl-A</b> | This command will select all the text.                                                                                        |
| <b>Ctrl-V</b> | This command will, when there is text present in the clipboard, paste the clipboard contents at the current position.         |
| <b>Ctrl-C</b> | This command will, when there is a selection, copy the selected text to the clipboard.                                        |
| <b>Ctrl-X</b> | This command will, when there is a selection, copy the selected text to the clipboard and remove the selection from the text. |

- **Search commands**

|                 |                                                                                                                                 |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------|
| <b>Ctrl-F3</b>  | This command will find the next occurrence of the word under the caret. When the next occurrence is found, it is selected.      |
| <b>F3</b>       | This command will find the next occurrence of the current search pattern. When the search pattern is found, it is selected.     |
| <b>Shift-F3</b> | This command will find the previous occurrence of the current search pattern. When the search pattern is found, it is selected. |
| <b>Ctrl-G</b>   | This command will show the <i>goto</i> dialog.                                                                                  |
| <b>Ctrl-F</b>   | This command will show the <i>find</i> dialog.                                                                                  |
| <b>Ctrl-H</b>   | This command will show the <i>replace</i> dialog.                                                                               |

- **File commands**

|                     |                                              |
|---------------------|----------------------------------------------|
| <b>Ctrl-N</b>       | Open a new file in a new tab window.         |
| <b>Ctrl-O</b>       | Open an existing file in a new tab window.   |
| <b>Ctrl-W</b>       | Close all files.                             |
| <b>Ctrl-S</b>       | This command will save the current file.     |
| <b>Ctrl-Shift-S</b> | Save all files.                              |
| <b>Ctrl-P</b>       | This command will open the print dialog.     |
| <b>Ctrl-I</b>       | Display the properties for the current file. |
| <b>Alt-F4</b>       | Exit the debugger.                           |

- **Bookmark commands**

|                      |                                                                                                                                                                           |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Ctrl-F2</b>       | This command will clear the bookmark on the current line if it is set, or set the bookmark if it is cleared.                                                              |
| <b>Shift-Ctrl-F2</b> | This command will clear all bookmarks.                                                                                                                                    |
| <b>F2</b>            | This command will place the caret on the next line which has a bookmark set. When there is no next line with a bookmark, the text is searched starting at the first line. |

**Shift-F2**

This command will place the caret on the previous line which has a bookmark set. When there is no previous line with a bookmark, the text is searched from the last line up again.

- **Breakpoint commands**

**Ctrl-F9**

This command will toggle a breakpoint on the current line.

**Ctrl-Shift-F9**

This command will clear all breakpoints.

- **Expression evaluation commands**

**Alt-F11**

Invoke the Evaluate Expression dialog.

**Alt-Shift-F11**

Invoke the Evaluate Expression dialog for the current selection. If no text is selected, evaluate the current line.

You can select the result and copy it to the clipboard.

## 6 Troubleshooting

- [Troubleshooting Service and Support](#)
- [Supported Platforms](#)
- [Help File](#)
- [Error Messages](#)
- [Registration](#)

### 6.1 Troubleshooting, Service & Support

If you need help with ***Take Command***, we encourage you to review our documentation and then contact us for assistance if required.

If you need help with sales, ordering, or registration keys, please contact our Sales and Customer Service department. See [Contacting JP Software](#) for our email address, mail address, and telephone numbers. Note that Sales and Customer service staff cannot assist you with technical problems and conversely Technical Support representatives cannot answer your sales or registration questions.

If you need technical support for ***Take Command***, review the [Technical Support](#) information section, which tells you what we need to know to provide you with accurate and timely support, then contact us via one of the methods described there. In most instances, our Online Support Forum is the fastest and most efficient way to address your technical questions and concerns.

Note that we do not provide support for **TCC/LE**. If you have questions about **TCC/LE**, post them on the [TCC/LE forum](#) on our web site.

#### 6.1.1 Technical Support

##### Support Plans

Standard, no-charge support is available electronically through our [Support Forums](#) (see below). We also offer a paid support option which includes automatic upgrades and support by private email or telephone. For complete details on all support options, including plans currently offered and support terms and conditions, see our web site at <http://jpsoft.com/>.

Before you contact Technical Support, please review the [What Information do we need?](#) section which outlines the basic data we need to best address your questions and concerns.

### Online Support

The primary venue for Technical Support is via our free online Support Forums, where our support personnel can read and respond to your messages, and other users can participate in and benefit from the exchange. The Forums are a lively community frequented by a number of experienced and helpful users. JP Software representatives read every Forum message and respond as promptly as reasonably possible whenever appropriate.

If you have any kind of Internet access, even if only email, chances are you can use the Forums which we make accessible as a mailing list and a set of web pages. Forum members must provide a valid email address and a full name to be able to post, but you do not need to join or provide any information to simply visit or search the Forum. For complete details and direct access links see the support area of our web site at <http://jpsoft.com/>.

A number of other support resources are available from our web site, including documentation files, technical tips and discussions, other technical information, and links to other sites. We update this information regularly, and we encourage you to check the Technical Support area of the web site to see if the information there will address any questions you have.

If you are unable to gain access to the forum, or you need to include confidential information in your support request, contact us via email at [support@jpsoft.com](mailto:support@jpsoft.com) and we will assist you in resolving the problem with forum access, or assist you with your request privately if appropriate. Please do not use that address for standard support questions which can be posted on the forum.

If you are a paid support customer you should use the online Support Forums for routine questions. To create a private support incident refer to the materials sent to you with your subscription for contact information, or email [priority\\_support@jpsoft.com](mailto:priority_support@jpsoft.com) and include your support ID (mail to this address may not be answered if it does not include a valid support ID).

### What Information do we need?

Before contacting us for support, please check this help file and other documentation for answers to your question. If you can't find what you need, try the Index. If you're having trouble getting **Take Command** to run properly, review the information on [Error Messages](#), and look through the Support Forum for any last-minute information.

If you need help with sales, ordering, registration keys, or other similar non-technical issues please contact our Sales and Customer Service department. Technical Support will not be able to assist you with those matters. Conversely, Customer Service is not equipped to answer your technical questions. See [Contacting JP Software](#) for our addresses.

Regardless of how you contact us for support, we can do a much better job of assisting you if you can give us some basic information, separate from your interpretations of or conclusions about the problem. Remember that we know NOTHING about your system or configuration unless you tell us, and we can't always make accurate guesses if you don't. The first four items listed below are essential for us to be able to understand and assist you with your problem:

- **What environment are you working in?** This includes the operating system version you are using, the version of the JP Software product involved, and related information such as network

connections and the name and version number of any other software which appears to be involved in the problem. Use the [VER/R](#) command to determine the **Take Command** version and operating system version. This item is essential! Every question posted on the Forum should include a brief identification such as "Take Command 16.0.25 under Windows 7 x64" or something similar.

- **What exactly did you do?** A concise description of what steps you must take to make the problem appear is much more useful than a long analysis of what might be happening. In most cases, posting the exact command line(s) giving you trouble is the simplest approach.
- **What did you expect to happen?** Tell us the result you expected from the command or operation in question, so that we understand what you are trying to do. Something that seems "obvious" to you might not be so to others. For example, tell us "I was expecting the file name to be in upper case" or a similar brief explanation.
- **What actually happened?** At what point did the failure occur? If you saw an error message or other important or unusual information on the screen, what **exactly** did it say? Don't simply tell us "it didn't work". For example, if you were expecting output from a command and saw none, at least tell us that much.
- **Briefly, what techniques did you use to try to resolve the problem?** What results did you get? One technique that tends to solve many problems is to review the help for the command or feature in question and try it with the documented exact correct syntax, as opposed to some undocumented alternative.
- **If the problem seems related to startup and configuration issues,** what are the contents of any startup files you use (such as [TCSTART](#) or [TCEXIT](#), and the .INI file), any batch files they call, and any alias or environment variable files they load?
- **Can you repeat the problem or does it occur randomly?** If it's random, does it seem related to the programs you're using when the problem occurs? Random or occasional problems are very difficult to diagnose. Do your best to determine some sort of pattern or sequence of events that triggers the problem. If you can't reproduce it, chances are we won't be able to either. Note that mysterious unexplainable problems often permanently disappear after simply reloading the program or even rebooting the system.
- If **Take Command** or **TCC** experience an unrecoverable failure, they will save the error information to a file called **TCMD.GPF** (which will be in the same directory as **TCMD.INI**). They will also send the **TCMD.GPF** file to JP Software at [support@jpsoft.com](mailto:support@jpsoft.com). If you can reproduce the error, please email us with the steps necessary to reproduce it and attach the **TCMD.GPF** file to your email; it will make it much easier for us to reproduce and fix the problem.

## 6.1.2 Contacting JP Software

You can contact JP Software at the following addresses and numbers. Our normal business hours are 9:00 AM to 5:00 PM weekdays, Eastern US time (except holidays).

Address: **JP Software Inc.  
P.O. Box 328  
Chestertown, MD 21620  
USA**

Phone: 800-595-8197  
Fax: 800-595-8197 (rings through to fax)

- Online: Web site: <http://jpsoft.com/>  
[Sales / Customer Service](#)
- Technical Support: Standard (no-charge) support: Available via our online [Support Forum](#), accessible from the support area of our web site.
- See [Technical Support](#) for additional details, and for information on paid support options.

**Note:** Our server implements anti-spam measures. Please make sure you are using the correct address with appropriate subject line and contents, else we might not receive your email message.

## 6.2 Supported Platforms

**Take Command** is a 32-bit or 64-bit GUI application.

**TCC** and **TCC/LE** are 32-bit or 64-bit console (character-mode) applications.

All are designed to run under Windows Vista, Windows 2008, Windows 7, Windows 8, and Windows 2012.

The 32-bit and 64-bit versions of **Take Command** and **TCC** are identical in features. The 64-bit versions are slightly faster and don't have to deal with the directory and registry remapping that Windows does with 32-bit apps running in 64-bit Windows. The 64-bit versions will also use slightly more memory.

## 6.3 Help File

The installers for **Take Command** and **TCC/LE** include **TCMD.CHM**, a help file in Windows' compiled HTML format. That file includes description and syntax for all commands, variables and functions, as well as reference information to assist you in installing and using **Take Command** and developing batch files, aliases and functions. **Take Command** uses the default Windows help system to display the contents of **tcmd.chm**. Under most configurations, Windows will remember the last used settings (window size, tab selected, etc.) for that file. Once you've started the help system with **F1** or the **HELP** command, you can use standard Windows HTML Viewer (HH.EXE) keystrokes to navigate. For more information, see your Windows documentation.

The help is also available on our web site as a PDF file.

You can request help at the prompt from the **Help** menu, by typing **HELP** (or **HELP** plus a command name), or by pressing the **F1** key at any time when **TCC** is accepting keyboard input at the prompt. If you use the **HELP** command by itself you will be taken to an introductory page, but if you follow the command with a topic name (e.g. **help copy**) you will see help on the requested topic if available.

If you type a command name on the line and then press **F1**, the help system will provide context sensitive help by using the first word on the line as the help topic. For example, if you press **F1** after entering each of the command lines shown below you will get the display indicated:

```
[c:\] Overview
[c:\] copy * a: Help on the COPY command
[c:\] c:\util\map "The page cannot be displayed"
```

You can use this feature to obtain help directly on any topic, not just on commands. All internal commands, internal variables, variable functions, and key mapping directives have their own topic,

allowing you to directly query, for example, **help @eval** (help for the @eval[] function) or **help \_disk** (help for the \_DISK internal variable).

You can also invoke help for the word immediately above (or immediately to the left of) the cursor by pressing the [Ctrl-F1](#) key. This feature is especially useful when you need the syntax for a variable function.

If the topic you seek is not listed, look for a suitable cross reference from the **Index** tab or use the **Search** tab. The topics you use most often can be stored and recalled through the **Favorites** tab.

#### Quick Syntax Help:

If you just need a quick reminder of an internal command's syntax, type the name of the command at the prompt, followed by a slash and a question mark **/?** For example:

```
copy /?
```

**TCC** will display the syntax and the valid options for the command.

If you are running **TCC** or **TCC/LE** in a **Take Command** tab window, and you enter an internal command name, **Take Command** will display quick help for that command on the status bar. If you move the mouse over the status bar, **Take Command** will display a more detailed tooltip help window.

## 6.4 Error Messages

This section lists error messages generated by **Take Command**, and includes a recommended course of action for most errors. If you are unable to resolve the problem after reviewing these help files, contact JP Software for [technical support](#).

Error messages relating to files are generally reports of errors returned by Windows. You may find some of these messages (for example, "Access denied") vague enough that they are not always helpful. **Take Command** includes the file name in file error messages, but is often unable to determine a more accurate explanation of these errors. The message shown is the best information available based on the error codes returned by Windows.

Not all errors potentially reported by Windows can be listed here. See [Windows System Errors](#) for examples of system errors returned in the [SYSERR](#) internal variable.

The following list includes most common error messages, in alphabetical order:

**Access denied:** You tried to write to or erase a read-only file, rename a file or directory to an existing name, create a directory that already exists, remove a read-only directory or a directory with files or subdirectories still in it, or access a file in use by another program in a multitasking system.

**Alias loop:** An alias refers back to itself either directly or indirectly (*i.e.*, `a = b = a`), or aliases are nested more than 16 levels deep. Correct your alias list.

**Already debugging a batch file:** You are attempting to invoke a nested instance of the Batch File Debugger ([BDEBUGGER](#)) while you are already in the debugger.

**Already excluded files:** You used more than one exclude range in a command. Combine the exclusions into a single range.

**Array variable is already defined:** You tried to create an array variable that already exists.

**Bad disk unit:** Generally caused by a disk drive hardware failure.

**Batch file missing:** *TCC* can't find the batch (.BTM or .CMD) file it was running. It was either deleted, renamed, moved, or the disk was changed. Correct the problem and rerun the file.

**Can't COPY or MOVE file to itself:** You cannot COPY or MOVE a file to itself. *TCC* attempts to perform full path and filename expansion before copying to help ensure that files aren't inadvertently destroyed.

**Can't create:** *TCC* can't create the specified file. The disk may be full or write protected, or the file already exists and is read-only, or the root directory is full.

**Can't delete:** *TCC* can't delete the specified file or directory. The disk is probably write protected.

**Can't end current process:** You attempted to terminate *TCC* with a [TASKEND](#) command. TASKEND can only be used to end other processes; to terminate *TCC*, use the [EXIT](#) command.

**Can't get directory:** *TCC* can't read the directory. The disk drive is probably not ready.

**Can't make directory entry:** *TCC* can't create the filename in the directory. This is usually caused by a full root directory. Create a subdirectory and move some of the files to it.

**Can't open:** *TCC* can't open the specified file. Either the file doesn't exist or the disk directory or File Allocation Table is damaged.

**Can't query key type:** The key name supplied to @REGQUERY refers to a key with a type that @REGQUERY does not support. See [@REGQUERY](#) for a list of supported key types.

**Can't remove current directory:** You attempted to remove the current directory, which Windows does not allow. Change to the parent directory and try again.

**CD-ROM door open or CD-ROM not ready:** The CD-ROM drive door is open, the power is off, or the drive is disconnected. Correct the problem and try again.

**CD-ROM not High Sierra or ISO-9660:** The CD-ROM is not recognized as a data CD (it may be a music CD). Put the correct CD in the drive and try again.

**Clipboard is empty or not text format:** You tried to retrieve some text from the Windows clipboard, but there is no text available. Correct the contents of the clipboard and try again.

**Clipboard is in use by another program:** *Take Command* could not access the Windows clipboard because another program was using it. Wait until the clipboard is available, or complete any pending action in the other program, then try again.

**Command line too long:** A single command or the entire command line exceeded the maximum [allowable length](#) (including during alias, variable, or function expansion). Reduce the complexity of the command or use a batch file. Also check for an alias which refers back to itself either directly or indirectly.

**Command only valid in batch file:** You have tried to use a batch file command, like DO or GOSUB, from the command line or in an alias. A few commands can only be used in batch files (see the individual

commands for details).

**Contents lost before copy:** COPY was appending files, and found one of the source files is the same as the destination. That source file is skipped, and appending continues with the next file.

**Data error:** Windows can't read or write properly to the device. On a floppy drive, this error is usually caused by a defective floppy disk, dirty disk drive heads, or a misalignment between the heads on your drive and the drive on which the disk was created. On a hard drive, this error may indicate a drive that is too hot or too cold, or a hardware problem. Retry the operation; if it fails again, correct the hardware or diskette problem.

**Directory stack empty:** [POPD](#) or [DIRS](#) can't find any entries in the directory stack.

**Disk is write protected:** The disk cannot be written to. Check the disk and remove the write-protect tab or close the write- protect window if necessary.

**Divide by zero:** The command or function you used tried to do a division by zero. If the data causing the problem is from your own input or batch file, change the input to avoid the divide by zero condition. If the data was generated internally by **Take Command**, contact JP Software for assistance.

**Drive not ready; close door:** The removable disk drive door is open. Close the door and try again.

**Duplicate redirection:** You tried to redirect standard input, standard output, or stand error more than once in the same command. Correct the command and try again.

**Error in command line directive:** You used the [//inline](#) option to place an .INI directive on the [startup](#) command line, but the directive is in error. Usually a more specific error message follows, and can be looked up in this list.

**Error on line [nnnn] of [filename]:** There is an error in your [.INI file](#). The following message explains the error in more detail. Correct the line in error and restart **TCC** for your change to take effect.

**Error reading:** Windows experienced an I/O error when reading from a device. This is usually caused by a bad disk, a device not ready, or a hardware error.

**Error writing:** Windows experienced an I/O error when writing to a device. This is usually caused by a full disk, a bad disk, a device not ready, or a hardware error.

**Exceeded batch nesting limit:** You have attempted to nest batch files more than 10 levels deep.

**Exceeded the maximum number of simultaneous monitors:** You have attempted to create more than 100 monitoring functions.

**File Allocation Table bad:** Windows can't access the FAT on the specified disk. This can be caused by a bad disk, a hardware error, or an unusual software interaction.

**File association not found:** The [ASSOC](#) command could not find a file association for the specified extension in the Windows registry.

**File exists:** The requested output file already exists, and **TCC** won't overwrite it.

**File not found:** **TCC** couldn't find the specified file. Check the spelling and path name.

**File type not found:** The [FTYPE](#) command could not find the specified file type in the Windows registry.

**General failure:** This is usually a hardware problem, particularly a disk drive failure or a device not properly connected to a serial or parallel port. Try to correct the problem, or reboot and try again. See also: [Data error](#) above.

**Include file not found:** You used the Include directive in the [.INI file](#), but the file you specified was not found or could not be opened.

**Include files nested too deep:** You used the Include directive in the [.INI file](#), and attempted to nest include files more than three levels deep.

**Infinite COPY or MOVE loop:** You tried to COPY or MOVE a directory to one of its own subdirectories and used the /S switch, so the command would run forever. Correct the command and try again.

**Input and output files must have different names:** ([BATCOMP](#)) You are attempting to compress a file to itself.

**Input file is already compressed:** ([BATCOMP](#)) You are attempting to compress a batch file that has already been compressed.

**Insufficient disk space:** COPY or MOVE ran out of room on the destination drive. Remove some files and retry the operation.

**Invalid array argument (out of bounds):** You tried to reference an array element that exceeded the array size.

**Invalid batch file:** The batch file is corrupted, or improperly [compressed](#), or encrypted. Retry with a new copy of the file.

**Invalid character value:** You gave an invalid value for a character directive in the [.INI file](#).

**Invalid choice value:** You gave an invalid value for a "choice" directive (one that accepts a choice from a list, like "Yes" or "No") in the [.INI file](#).

**Invalid color:** You gave an invalid value for a color directive in the [.INI file](#).

**Invalid count:** The character repeat count for [KEYSTACK](#) is incorrect.

**Invalid date:** An invalid date was entered. Check the syntax and reenter.

**Invalid directive name:** *Take Command* can't recognize the name of a directive in the [.INI file](#).

**Invalid drive:** A bad or non-existent disk drive was specified.

**Invalid key name:** You tried to make an invalid key substitution in the [.INI file](#), or you used an invalid key name in a keystroke [alias](#) or command. Correct the error and retry the operation.

**Invalid numeric value:** You gave an invalid value for a numeric directive in the [.INI file](#).

**Invalid parameter:** TCC didn't recognize a parameter. Check the syntax and spelling of the command you entered.

**Invalid path:** The specified path does not exist. Check the disk specification and/or spelling.

**Invalid path or file name:** You used an invalid path or filename in a directive in the [.INI file](#).

**Invalid time:** An invalid time was entered. Check the syntax and reenter.

**Keystroke substitution table full:** *TCC* ran out of room to store [keystroke substitutions](#) entered in the [.INI file](#). Reduce the number of key substitutions or contact JP Software or your dealer for assistance.

**Label not found:** A [GOTO](#) or [GOSUB](#) referred to a non-existent label. Check your batch file.

**Listbox is full:** There is no more room in the Find Files / Text dialog's results box. Use a more selective search, or use the FFIND command rather than the dialog.

**Missing close paren:** A [KEYSTACK](#) command is missing a closing parentheses around a character group. Correct the command.

**Missing ENDTEXT:** A [TEXT](#) command is missing a matching ENDTEXT. Check the batch file.

**Missing GOSUB:** *TCC* cannot perform the [RETURN](#) command in a batch file. You tried to do a RETURN without a [GOSUB](#), or your batch file has been corrupted.

**Missing SETLOCAL:** An [ENDLOCAL](#) was used without a matching [SETLOCAL](#).

**No aliases defined:** You tried to display aliases but no aliases have been defined.

**Not an array variable:** You tried to reference a non-existent array variable.

**No closing quote:** *TCC* couldn't find a second matching back quote [`] or double-quote ["] on the command line.

**No expression:** The expression passed to the [%@EVAL](#) variable function is empty. Correct the expression and retry the operation.

**No shared memory found:** The [SHRALIAS](#) command could not find any global alias list, history list, or directory history list to retain, because you executed the command from a session with local lists. Start *TCC* with at least one global list, then invoke SHRALIAS.

**No SMTP server:** [SENDMAIL](#) can't find an SMTP server. Check your INI file or mailer configuration (see SENDMAIL for additional details).

**Not a directory:** The name passed to [RD](#) is not a directory.

**Not an alias:** The specified alias is not in the alias list.

**Not in environment:** The specified variable is not in the environment.

**Not ready:** The specified device can't be accessed.

**Not same device:** This error usually appears in [RENAME](#). You cannot rename a file to a different disk drive.

**Out of function space:** You are attempting to create a [User-defined Function](#) that would require more resources than what your system makes available. Shorten the function definition or delete functions you no longer need.

**Out of memory:** *Take Command* or Windows had insufficient memory to execute the last command. Try to free some memory by closing other sessions. If the error persists, contact JP Software for assistance.

**Out of paper:** Windows detected an out-of-paper condition on one of the printers. Check your printer and add paper if necessary.

**Overflow:** An arithmetic overflow occurred in the [@EVAL](#) variable function. Check the values being passed to @EVAL.

**Read error:** Windows encountered a disk read error; usually caused by a bad or unformatted disk. See also: **Data error** above.

**Sector not found:** Disk error, usually caused by a bad or unformatted disk. See also: **Data error** above.

**Seek error:** Windows can't seek to the proper location on the disk. This is generally caused by a bad disk or drive. See also: **Data error** above.

**Sharing violation:** You tried to access a file in use by another program in a multitasking system or on a network. Wait for the file to become available, or change your method of operation so that another program does not have the file open while you are trying to use it.

**SHRALIAS already loaded:** You used the [SHRALIAS](#) command to load SHRALIAS.EXE, but it was already loaded. This message is informational and generally does not indicate an error condition.

**SHRALIAS not loaded:** You used the [SHRALIAS /U](#) command to unload SHRALIAS.EXE, but it was never loaded. This message is informational and may not indicate an error condition.

**String area overflow:** *TCC* ran out of room to store the text from string directives in the [.INI file](#). Reduce the complexity of the .INI file or contact JP Software for assistance.

**String too long:** You tried to put more than 2038 characters into the [KEYSTACK](#) buffer. Reduce the number of characters you are trying to send to the application at one time.

**Syntax error:** A command or [variable function](#) was entered in an improper format. Check the syntax and correct the error.

**Too many open files:** Windows has run out of file handles.

**Unbalanced parentheses:** The number of left and right parentheses did not match in an expression passed to the [@EVAL](#) variable function. Correct the expression and retry the operation.

**UNKNOWN\_CMD loop:** The [UNKNOWN\\_CMD alias](#) called itself more than ten times. The alias probably contains an unknown command itself, and is stuck in an infinite loop. Correct the alias.

**Unknown command:** A command was entered that *TCC* didn't recognize and couldn't find in the current search path. Check the spelling or PATH specification. You can handle unknown commands with the UNKNOWN\_CMD alias (see [ALIAS](#)).

**Unknown option name:** ([OPTION](#)) You are attempting to modify or display an invalid or unknown option name.

**Unknown process:** [TASKEND](#) cannot find the process you specified. If you are ending a process using the title you may need to use wildcards to get a match on the title string. Correct the command and try again.

**Variable loop:** A nested environment variable refers to itself, or variables are nested more than 16 deep. Correct the error and retry the command.

**Window title not found:** The [ACTIVATE](#) command could not find a window with the specified title. Correct the command or open the appropriate window and try again.

**Write error:** Windows encountered a disk write error; usually caused by a bad or unformatted disk. See also: **Data error** above.

## 7 Registration

There are no separate **trial** and **registered** versions of **Take Command**. Without registration, a trial version is fully functional for 30 days of use.

At any time you can apply your current personal registration information to a trial version in order to turn it into a registered product. Use the command [VER/R](#) from the **TCC** prompt to verify the status of the copy you are currently running. You can also view the [Help/About](#) menu entry in **Take Command**.

When you purchase a new or upgrade copy of **Take Command**, you will receive an email with your name and registration key. Start **Take Command**, click on the **Options** menu entry and then **Configure Take Command**. Select the **Register** tab and enter the registration information exactly as you received it in the email. Remember to save your registration key in a safe place in case you need to reinstall. If you have lost your registration key, you can request a replacement by contacting JP Software at [operations@jpsoft.com](mailto:operations@jpsoft.com).

When you enter your registration information, **Take Command** will save it in the Windows Registry unless the **Take Command** installation is on a removable drive. The drive must be identified by Windows as a removable drive (many USB external disk drives identify themselves to Windows as non-removable). If you are unsure, try running:

```
echo %@removable[d:]
```

where **d:** is the drive letter. If it returns 1, then it is a removable drive. If **Take Command** is installed on a removable drive, it will read / write its registration info to a key file in the **Take Command** installation directory. This allows you to move your registered copy of **Take Command** among different machines by installing it to a USB thumb drive. Note that you cannot install **Take Command** on a removable drive if you have already installed it elsewhere on your system -- the Windows Installer does not support installing multiple copies of the same program version on the same system.

(Windows 7/8 and Vista) If you are running an elevated administrator session, you can optionally check the "Register for All Users" option.

## 8 Reference

- [Updater](#)
- [Windows x64](#)
- [Plugins](#)
- [CMD Comparison](#)
- [CMD Compatibility](#)
- [Limits](#)
- [File Systems and File Name Conventions](#)
- [Regular Expression Syntax](#)
- [XML in TCC](#)
- [Miscellaneous Reference Information](#)
- [ANSI X3.64 Command Reference](#)
- [Colors, Color Names & Codes](#)

### 8.1 Updater

You can check for updates to **Take Command** with the Help / Check for Updates menu entry. (In **TCC**, you can also use the OPTION / Check for Updates tab.)

You can also automate updates by using the Updater tool. UPDATE.EXE is a small executable tool (located in your **Take Command** installation directory) whose role is to check for updates, inform the user of their presence and offer to download and install them. When launched, the Updater checks if a newer version of **Take Command** exists.

#### Updater options

You can also run the Updater from the command line or batch files. The Updater has the following command line options:

- /checknow - The Updater is launched, pops up a dialog box, checks for updates and automatically informs the user that new updates are available. If no updates are available, the Updater will exit immediately.
- /silent - The Updater will search silently for updates at the interval specified by the user. The search interval can be specified in the Configuration dialog. By default it is the value you specified in the "Check Frequency" field (Updater Page). If the check frequency has not passed or there are no updates available, the Updater will exit immediately.
- /silentall - The Updater will search silently for updates and automatically install all updates. This has the same effect as the /silent option if the user has selected the "Check and automatically install all updates" option in the configuration dialog. If the check frequency has not passed or there are no updates available, the Updater will exit immediately.
- /configure - the Configure dialog will be displayed allowing the configuration of the Updater. In this dialog the user can specify the download folder, the check frequency, enable or disable the automatically update option.

### 8.2 Windows x64

**Take Command**, **TCC**, and **TCC/LE** come in both 32-bit (x86) and 64-bit (x64) versions. The 32-bit versions will run on either 32-bit or 64-bit versions of Windows; the 64-bit versions will only run on 64-bit Windows.

There are a few differences between the 32 and 64-bit **Take Command / TCC** versions as a result of

third-party module differences. Perl and Python are available from ActiveState in 64-bit versions ([www.activestate.com](http://www.activestate.com)), but Tcl is only available in a 32-bit version. (You can get the source code for Tcl and with a bit of work build an x64 version.)

## 8.3 Plugins

**TCC** plugins are user-written DLL's that allow you to write your own internal variables, variable functions, and internal commands, and have **TCC** load them at startup. Plugin names will override existing names, so you can extend and/or replace internal variables and commands. When **TCC** starts, it will automatically load any plugins in the default directory (the subdirectory PLUGINS\ in the **TCC** installation directory). The plugins will be loaded before the startup file ([TCSTART](#)) are executed.

You can also write keystroke plugins that will be called for every keystroke entered at the command line. A keystroke plugin can perform actions when a specific key is entered, or even change the key before passing it back to the command processor.

Plugins can be written in any language that can create a Windows DLL. The **TCC** plugin SDK has samples for Visual C++ and Delphi. The SDK is available on our web site at <http://jpssoft.com/downloads/sdk/sdk.zip>.

### Plugin Syntax:

```
// PluginInfo structure - returned by plugin in response to GetPluginInfo() call from
// command processor
// Note that the strings should all be Unicode; if your Plugin is compiled for ASCII
// you'll need to use
// the MultiByteToWideChar API to convert the strings before passing them back to
// TCC
typedef struct {
 TCHAR *pszDll; // name of the DLL
 TCHAR *pszAuthor; // author's name
 TCHAR *pszEmail; // author's email
 TCHAR *pszWWW; // author's web page
 TCHAR *pszDescription; // (brief) description of plugin
 TCHAR *pszFunctions; // comma-delimited list of functions in the
 // plugin (leading _ for internal vars, @
for
 // var funcs, * for keystroke function,
 // otherwise it's a command)
 int nMajor; // plugin's major version #
 int nMinor; // plugin's minor version #
 int nBuild; // plugin's build #
 HMODULE hModule; // module handle
 TCHAR *pszModule; // module name
} PLUGININFO, *LPPLUGININFO;

// structure passed to plugin functions to monitor keystrokes. A
// keystroke function can be named anything, but must prefix a
// * to its name in the function list (pszFunctions, above).
// If the keystroke plugin handled the keystroke and doesn't want
// pass it back to TCC, it should set nKey = 0
// The command processor will call the keystroke function with all
// parameters set to 0 just before accepting input for each new
// command line.
// The string pointers are Unicode
typedef struct {
 int nKey; // key entered
```

```

int nHomeRow; // start row
int nHomeColumn; // start column
int nRow; // current row in window
int nColumn; // current column in window
LPTSTR pszLine; // command line
LPTSTR pszCurrent; // pointer to position in line
int fRedraw; // if != 0, redraw the line
} KEYINFO, *LPKEYINFO;

__declspec(dllexport) BOOL WINAPI InitializePlugin(void); // called
by command processor after loading all plugins
__declspec(dllexport) LPPLUGININFO WINAPI GetPluginInfo(HMODULE hModule); // called
by command processor to get information from plugin, primarily for the names of
functions & commands
__declspec(dllexport) BOOL WINAPI ShutdownPlugin(BOOL bEndProcess); // called
by command processor when shutting down
 // if
bEndProcess = 0, only the plugin is being closed
 // if
bEndProcess = 1, the command processor is shutting down

```

The functions listed in "pszFunctions" and called by TCC need to be in the format:

DLLExports INT WINAPI MyFunctionName( LPTSTR pszArguments );

Internal variable names in pszFunctions (and their corresponding functions) must begin with an underscore ('\_').

Variable function names in pszFunctions must begin with an @; the corresponding function must be prefixed by "f\_".

(This allows variable functions to have the same name as internal commands.)

For example:

pszFunctions = "reverse,@reverse"

Entering the name "reverse" on the command line will invoke the command reverse()

Entering the name "@reverse[]" on the command line will invoke the variable function f\_reverse()

Variable function names are limited to a maximum of 31 characters.

Internal command names are any combination of alphanumeric characters (maximum 12 characters).

### **Calling the Plugin:**

For internal variables, pszArguments is empty (for output only)

For variable functions, pszArguments passes the argument(s) to the plugin function

For internal commands, pszArguments is the command line minus the name of the internal command

### **Returning from the Plugin:**

For internal variables and variable functions, copy the result string over pszArguments. The maximum string length for internal variables and variable functions is 2K (2047 characters + null byte).

Internal variables have no meaningful integer return value. For variable functions, the integer return can be:

```
0 = success
< 0 = failure; error message already displayed by the Plugin function
> 0 = failure; error value should be interpreted as a system error and
 displayed by 4NT / TC
```

There is a special return value (0xFEDCBA98) that tells the parser to assume that the plugin decided not to handle the variable/function/command. The parser then continues looking for a matching internal, then external. Note that you can use this return value to have your plugin modify the command line and then pass it on to an existing internal variable/function/command!

For internal commands, return the integer result (anything left in pszArgument will be ignored)

#### **Exception Handling:**

TCC will trap any exceptions occurring in the plugin, to prevent the plugin from crashing the command processor. An error message will be displayed and the plugin will return an exit code = 2.

#### **Take Command Interface:**

Added some new API functions for manipulating the directory history and command history:

DirHistoryStart(void) - returns a pointer to the beginning of the directory history

HistoryStart(void) - returns a pointer to the beginning of the command history

DeleteFromHistory( LPTSTR IpszLine ) - deletes the line from the command history (this is a pointer to the line to be deleted, not a line to be matched!)

If the user tries to display online help with HELP, F1 or Ctrl-F1, TCC will check for a plugin variable, variable function, or command, and if the name matches search for, load and execute a "Help" function in the plugin. The plugin is responsible for displaying its own help. The "Help" function should NOT appear in the plugin's comma-delimited function list in pszFunctions. Help should return 1 if it displayed help (or if it doesn't want TCC to try to display help for this topic). The syntax of the Help function in the plugin should be:

Help( LPTSTR pszName );

If Take Command wants to display usage text, TCC will check for a plugin command, and if the name matches search for, load and execute a "Usage" function in the plugin. The plugin is responsible for displaying its own help. The "Usage" function should NOT appear in the plugin's comma-delimited function list in pszFunctions. The plugin should return a multi-line string containing the command syntax. The first line (terminated by a \r) is displayed in the Take Command status bar. The entire string is displayed as a tooltip popup when the mouse hovers over the status bar message. Usage should return 1 if it wrote something to pszUsage (or if it doesn't want TCC to try to display a usage string). The syntax of the Usage function in the plugin should be:

Usage( LPTSTR pszName, LPTSTR pszUsage );

The TakeCommandIPC function allows plugins to communicate with the controlling Take Command instance. The syntax is:

`__declspec(dllexport) int TakeCommandIPC( LPTSTR pszCommand, LPTSTR pszArguments );`

The supported commands are:

**HWND**

Returns the Take Command window handle in `pszArguments`

**TCTAB**

Returns 1 if the process ID in `pszArguments` is running in a TC window

**TCTABS**

Returns the number of Take Command tab windows

**HVIEW**

Returns the handle of the active tab window in `pszArguments`

**HELP**

Displays the Take Command help for the topic in `pszArguments`

**USAGE**

Display the usage message in `pszArguments` in the status bar. The first line (up to the first CR) is displayed in the status bar; the rest is displayed in the tooltip if you hover the mouse over the status bar.

**STATUSBAR**

Display the message in `pszArguments` in the status bar

**TCTOOLBAR**

Update the Take Command tab toolbar with `pszArguments`

**TCFILTER**

Return the selected filter in the list view in `pszArguments`

**TCFILTER\_CMD**

Set the selected filter in the list view to the value in `pszArguments`

**CDD**

Change the folder and list view to the directory in `pszArguments`

**TCFOLDER**

Return the selected folder in the Folders tree control in `pszArguments`

**SHORTCUT**

Return the name of the shortcut that started Take Command in `pszArguments`

**SELECTED**

Return the currently selected text in `pszArguments`

**SELECT**

Mark the selection specified in `pszArguments` (top, left, bottom, right) in Take Command

**START**

Attach a hidden console window whose hex PID is in `pszArguments`

**ACTIVATE**

Activate the window whose handle is in pszArguments

**WINDOW**

Has a number of arguments (specified in pszArguments) to control the Take Command window:

- MAX
- MIN
- HIDE
- RES
- TRAY
- TRANS=n
- FLASH=n
- DETACH n (where n is the PID of the process to detach)
- TOPMOST
- NOTOPMOST
- TOP
- BOTTOM

**ArrayVariables**

Plugins can access TCC array variables directly through the ArrayVariables array. See TakeCmd.h in the SDK for details.

## 8.4 CMD.EXE Comparison

The comparison of commands available is based on the version of **CMD** included with Windows 7.

If the CMD command name matches an internal **TCC** command, the **TCC** command is almost always substantially enhanced.

- [TCC, TCC/LE, and CMD commands](#)
- [Command line editing](#)
- [Filename completion](#)
- [Command completion](#)
- [Redirection](#)
- [Wildcards](#)
- [Built-In Variables](#)
- [Unique TCC features](#)

***TCC, TCC/LE, and CMD commands***

| <b>TCC</b>                | <b>TCC/LE</b> | <b>CMD</b> |
|---------------------------|---------------|------------|
| ?                         | Y             |            |
| <a href="#">ACTIVATE</a>  | Y             |            |
| <a href="#">ALIAS</a>     | Y             |            |
| <a href="#">ASSOC</a>     | Y             | Y          |
| <a href="#">ASSOCIATE</a> |               |            |
| <a href="#">ATTRIB</a>    | Y             | *          |
| <a href="#">BATCOMP</a>   |               |            |
| <a href="#">BDEBUGGER</a> |               |            |
| <a href="#">BEEP</a>      | Y             |            |

|                 |   |   |
|-----------------|---|---|
| BREAK           | Y | Y |
| BREAKPOINT      |   |   |
| CALL            | Y | Y |
| CANCEL          | Y |   |
| CD / CHDIR      | Y | Y |
| CDD             | Y |   |
| CHCP            | Y | * |
| CLIPMONITOR     |   |   |
| CLS             | Y | Y |
| COLOR           | Y | Y |
| COPY            | Y | Y |
| DATE            | Y | Y |
| DATEMONITOR     |   |   |
| DEBUGMONITOR    |   |   |
| DEBUGSTRING     |   |   |
| DEFER           |   |   |
| DEL / ERASE     | Y | Y |
| DELAY           | Y |   |
| DESCRIBE        | Y |   |
| DESKTOP         |   |   |
| DETACH          |   |   |
| DIR             | Y | Y |
| DIRHISTORY      | Y |   |
| DIRS            | Y |   |
| DISKMONITOR     |   |   |
| DO              | Y |   |
| DRAWBOX         | Y |   |
| DRAWLINE        | Y |   |
| DRAWVLINE       | Y |   |
| ECHO            | Y | Y |
| ECHOERR         | Y |   |
| ECHOS           | Y |   |
| ECHOSERR        | Y |   |
| ECHOX           |   |   |
| ECHOXERR        |   |   |
| EJECTMEDIA      | Y |   |
| ENDLOCAL        | Y | Y |
| ESET            | Y |   |
| EVENTLOG        |   |   |
| EVENTMONITOR    |   |   |
| EVERYTHING      |   |   |
| EXCEPT          | Y |   |
| EXIT            | Y | Y |
| FFIND           | Y |   |
| FIREWIREMONITOR |   |   |
| FOLDERMONITOR   |   |   |
| FONT            |   |   |
| FOR             | Y | Y |
| FREE            | Y |   |
| FTYPE           | Y | Y |
| FUNCTION        |   |   |
| GLOBAL          | Y |   |
| GOSUB           | Y |   |

|                |   |   |
|----------------|---|---|
| GOTO           | Y | Y |
| GZIP           |   |   |
| HEAD           |   |   |
| HELP           | Y | * |
| HISTORY        | Y |   |
| IDE            |   |   |
| IF             | Y | Y |
| IFF            | Y |   |
| IFTP           |   |   |
| INKEY          | Y |   |
| INPUT          | Y |   |
| JABBER         |   |   |
| KEYBD          | Y |   |
| KEYS           | Y | Y |
| KEYSTACK       | Y |   |
| LIST           | Y |   |
| LOADBTM        |   |   |
| LOADMEDIA      | Y |   |
| LOG            | Y |   |
| LUA            |   |   |
| MD / MKDIR     | Y | Y |
| MEMORY         | Y |   |
| MKLINK         | Y | Y |
| MKLNK          | Y |   |
| MOVE           | Y | Y |
| MSGBOX         | Y |   |
| NETMONITOR     |   |   |
| ON             | Y |   |
| OPTION         | Y |   |
| OSD            |   |   |
| PATH           | Y | Y |
| PAUSE          | Y | Y |
| PDIR           | Y |   |
| PLAYAVI        |   |   |
| PLAYSOUND      |   |   |
| PLUGIN         | Y |   |
| POPD           | Y | Y |
| POSTMSG        |   |   |
| PRINT          |   |   |
| PRIORITY       |   |   |
| PROCESSMONITOR |   |   |
| PROMPT         | Y | Y |
| PUSHD          | Y | Y |
| QUERYBOX       | Y |   |
| QUIT           | Y |   |
| RD / RMDIR     | Y | Y |
| REBOOT         | Y |   |
| RECYCLE        |   |   |
| REM            | Y | Y |
| REN / RENAME   | Y | Y |
| RESOLUTION     |   |   |
| RETURN         | Y |   |
| REEXEC         |   |   |

|                |   |   |
|----------------|---|---|
| RSHELL         |   |   |
| SCREEN         | Y |   |
| SCREENMONITOR  |   |   |
| SCRIPT         |   |   |
| SCRPUT         | Y |   |
| SELECT         | Y |   |
| SENDHTML       |   |   |
| SENDMAIL       |   |   |
| SERVICEMONITOR |   |   |
| SERVICES       |   |   |
| SET            | Y | Y |
| SETARRAY       |   |   |
| SETDOS         | Y |   |
| SETERROR       |   |   |
| SETLOCAL       | Y | Y |
| SHIFT          | Y | Y |
| SHORTCUT       |   |   |
| SHRALIAS       | Y |   |
| SMPP           |   |   |
| SNMP           |   |   |
| SNPP           |   |   |
| START          | Y | Y |
| STATUSBAR      |   |   |
| SWITCH         | Y |   |
| SYNC           |   |   |
| TAIL           |   |   |
| TAR            |   |   |
| TASKBAR        |   |   |
| TASKDIALOG     |   |   |
| TASKEND        |   |   |
| TASKLIST       |   |   |
| TCDIALOG       |   |   |
| TCFILTER       | Y |   |
| TCTOOLBAR      | Y |   |
| TEE            | Y |   |
| TEXT           | Y |   |
| TIME           | Y | Y |
| TIMER          | Y |   |
| TITLE          | Y | Y |
| TOUCH          | Y |   |
| TPIPE          |   |   |
| TRANSIENT      |   |   |
| TREE           | Y | * |
| TRUENAME       | Y |   |
| TYPE           | Y | Y |
| UNALIAS        | Y |   |
| UNFUNCTION     |   |   |
| UNGZIP         |   |   |
| UNSET          | Y |   |
| UNSETARRAY     |   |   |
| UNTAR          |   |   |
| UNZIP          |   |   |
| USBMONITOR     |   |   |

|           |   |   |
|-----------|---|---|
| VBEEP     |   |   |
| VER       | Y | Y |
| VERIFY    | Y | Y |
| VIEW      |   |   |
| VOL       | Y | Y |
| VSCRPUT   | Y |   |
| WAKEONLAN |   |   |
| WEBFORM   |   |   |
| WEBUPLOAD |   |   |
| WHICH     | Y |   |
| WINDOW    | Y |   |
| WMIQUERY  |   |   |
| Y         | Y |   |
| ZIP       |   |   |
| ZIPSFX    |   |   |
| 7UNZIP    |   |   |
| 7ZIP      |   |   |

\* This is an internal command in **TCC** but an external command in CMD.

### Command line editing

**TCC** offers vastly more sophisticated command line editing capabilities; see [Command Line Editing](#) for details.

### Filename completion

CMD has a simple filename completion (with the tab key); **TCC** offers many more options, including server and sharename completion, [customizable completion](#) and (optional) popup window selection. See [Filename Completion](#) and [Filename Completion Window](#) for more details.

### Command history

CMD has simple (optional) command history recall. **TCC** offers many more options, including loading and saving history lists, editing and moving commands in the list, searching for matching commands, and a popup command history window.

### Redirection

In addition to the CMD <, > and |, **TCC** allows you to also redirect standard error, combine standard output and standard error, protect existing files from being overwritten by redirection, and redirect standard input using "here-documents". See [Redirection](#) for more details.

### Wildcards

CMD only supports the ? and \* wildcards in filenames. **TCC** adds character sets and regular expressions, and also supports wildcards in pathnames. See [Wildcards](#) for more details.

### Built-In Variables

CMD has a few built-in variables (i.e., which are treated as environment variables but which do not exist in the environment):

**CD** - current directory

**CMDCMDLINE** - command line that started CMD

**CMDEXTVERSION** - the command extensions internal version number

**DATE** - the current date (in the default short format)

**RANDOM** - a random number between 0 and 32767

**TIME** - current time

**TCC** supports all of these built-in variables. (In **TCC**, **CMDEXTVERSION** will always return 2.) **TCC** also includes 185+ additional [internal variables](#), 330+ [variable functions](#), and 60+ [command variables](#).

### Unique **TCC** features

**TCC** includes many more features not in CMD, including:

[\*\*Batch debugger\*\*](#)

[\*\*Aliases\*\*](#)

[\*\*Internal functions\*\*](#)

[\*\*User defined functions\*\*](#)

[\*\*File selection\*\*](#)

[\*\*File Ranges\*\*](#)

[\*\*Conditional Commands\*\*](#)

[\*\*Internet access and email\*\*](#)

[\*\*OpenAFS support\*\*](#)

[\*\*ANSI X3.64 support\*\*](#)

[\*\*Directory navigation\*\*](#) and [\*\*Directory History\*\*](#)

[\*\*Histories and Logs\*\*](#)

[\*\*Intersession sharing\*\*](#)

[\*\*Lua\*\*](#), [\*\*Perl\*\*](#), [\*\*Python\*\*](#), [\*\*REXX\*\*](#), [\*\*Ruby\*\*](#), and [\*\*Tcl\*\*](#) support

## 8.5 CMD Compatibility

We try to keep **TCC** as compatible as possible with CMD, given the limitations and bugs in CMD, the variances in CMD in different versions of Windows, and the thousands of additional features available in

**TCC.** On rare occasions, you may find batch files that exploit undocumented features (or bugs) in CMD (or are simply badly written) that don't work in **TCC**. In almost all of those cases, **TCC** will run those batch files if you set the appropriate compatibility options.

There are two options you should set if you regularly run batch files created for CMD:

[\*\*OPTION\*\*](#) / Startup / Duplicate CMD.EXE bugs (This is the default, and tells **TCC** to duplicate two bugs in CMD's IF command parsing.)

[OPTION](#) / Startup / CMD.EXE delayed expansion (If you have this startup option set for your CMD environment.)

If you **only** run batch files created for CMD, you should also set:

[\*\*CMDVariables\*\*=YES](#)

in your TCMD.INI file. **WARNING:** This means you won't be able to run any batch files written for **TCC**, which only requires a single leading % for variables.

There are also some **TCC** features that might on very rare occasions cause conflicts with CMD batch files:

Enable "OPTION / Startup / Search for SFNs". (Definitely **not** recommended unless you want some potentially unpleasant results when you're copying, moving, or deleting files, but it \*is\* how CMD does it.)

Disable pseudovariable expansion (OPTION / Advanced / Special Characters)

[\*\*SETDOS\*\*](#) /X279 to disable nested aliases, quoting, and include lists.

Finally, if you want CMD-style command line editing (i.e., practically none), you can remove most of the **TCC** command line editing features with the OPTION / Command Line dialog.

## 8.6 Limits

### Length Limits (characters)

| entity                | name | value  | combined |
|-----------------------|------|--------|----------|
| environment variable  | 1023 | 32,767 | 32,767   |
| alias                 | 1023 | 32,767 | 32,767   |
| user defined function | 1023 | 32,767 | 32,767   |

| command type  | before expansion | after expansion |
|---------------|------------------|-----------------|
| command line  | 32,767           | 65,535          |
| command group | 32,767           | 65,535          |

### Nesting Limits

| command | depth    |
|---------|----------|
| CALL    | 32       |
| DO      | no limit |
| FOR     | no limit |

|                                          |          |
|------------------------------------------|----------|
| <a href="#">GOSUB</a> without parameters | no limit |
| <a href="#">GOSUB</a> with parameters    | 22       |
| <a href="#">SETLOCAL</a>                 | 16       |
| <a href="#">IFF</a>                      | no limit |

## Miscellaneous Limits (characters)

| entity                                    | limit   |
|-------------------------------------------|---------|
| character count in any function           | 32,767  |
| number of batch file parameters           | 4,095   |
| number of GOSUB parameters                | 255     |
| file name                                 | 32,767  |
| include list                              | 32,767  |
| single parameter                          | 32,767  |
| global alias list *                       | 262,144 |
| global function list *                    | 131,072 |
| key substitution table                    | 128     |
| directory stack ( <a href="#">PUSHD</a> ) | 4,095   |

\* The global alias list and global function list sizes may be increased with the [AliasSize](#) and [FunctionSize](#).INI directives.

## 8.7 File Systems & File Names

The unique name of any file is composed of a drive letter, a directory path, and a filename. In Windows, each of these parts of the file's name is case insensitive; you can mix upper and lower case letters in any way you wish. (Note that when accessing Linux / UNIX FTP servers, the filenames **are** case sensitive.)

The topics below are roughly divided according to the different parts of a file name, and cover the file system structure and naming conventions:

- [Drives and Volumes](#)
- [File Systems](#)
- [Directories and Subdirectories](#)
- [File Names](#)
- [File Attributes](#)
- [Time Stamps](#)
- [NTFS Streams](#)

### 8.7.1 Drives & Volumes

A **drive letter** designates which drive contains the file. In a file's full name, the drive letter is followed by a colon. Drive letters **A:** and **B:** are normally reserved for the floppy disk drives (now largely obsolete).

Normally, drive **C:** is the first (or only) hard disk drive. Most current operating systems can partition a large hard disk into multiple logical drives or volumes that are usually called **C:, D:, E:,** etc. Network systems (LANs) give additional drive letters to sections of the network file server drives. In addition, you can access network drives via their **UNC** (universal naming convention) name (e.g. `\data\vo\1\...`), without using a drive letter. See [File Systems](#) for more details.

Most systems also include optical drives (i.e. CD-ROM, CD-RW, and/or DVD). The optical drive is also

assigned a drive letter (or several letters, for changers), typically using letters beyond that used by the last hard disk in the system, but before any network drives.

For example, on a system with a large hard disk you might have **A:** and **B:** as floppy drives, **C:, D:, and E:** as parts of the hard disk, **F:** as a CD-ROM drive, **G:** as a DVD drive, and **H: and I:** as network drives.

Each volume is formatted under a particular file system; see [File Systems](#) for details. Additional information about disk files and directories is available under [Directories and Subdirectories](#), [File Names](#), and [File Attributes](#).

## 8.7.2 File Systems

**Take Command** uses only documented Windows APIs to access the file systems, so it works with any file system supported by Windows.

Additional information about disk files and directories is available under [Drives and Volumes](#), [Directories and Subdirectories](#), [File Names](#), and [File Attributes](#).

### Network File Systems

A network file system allows you to access files stored on another computer on a network, rather than on your own system. **Take Command** supports all network file systems which are compatible with the underlying operating system. The networking software used to access remote systems (such as UNIX, Linux, OS X, etc..) which use different file systems typically emulates one of the common Windows file systems. Those emulations do not always provide a perfect duplicate of some functions (attributes, timestamps, etc.), an issue unrelated to **Take Command**.

File and directory names for network file systems depend on both the "server" software running on the system that has the files on it, and the "client" software running on your computer to connect it to the network. However, they usually follow the rules described here.

Most network software maps unused drive letters on your system to specific locations on the network, and you can then treat the drive as if it were physically part of your local computer.

When you use a network file system, remember that the naming rules for files on the network may not match those on your local system. For example, your local system may support long filenames while the network server or client software does not, or vice versa. **Take Command** will usually handle whatever naming conventions are supported by your network software, as long as the network software accurately reports the types of names it can handle.

In rare cases, **Take Command** may not be able to report correct statistics on network drives (such as the number of bytes free on a drive). This is usually because the network file system does not provide complete or accurate information.

### Universal Naming Convention (UNC)

Some networks also support the Universal Naming Convention, which provides a common method for accessing files on a network drive without using a mapped drive letter. Names specified this way are called UNC names. They typically appear as `\server\path\filename`, where **server** is the name of the network server where the files reside, and the **path\filename** portion is a directory name and file name which follow the conventions described under [Directories](#).

**Take Command** also allows you to use UNC directory names when changing directories (see [Directory Navigation](#) for more details).

## OpenAFS

**Take Command** and **TCC** have built-in support for OpenAFS. (**TCC/LE** does not support OpenAFS.) The parser will recognize Linux-style AFS names (i.e., `/afs/athena/user`) and convert them to Windows-compatible names (i.e., `\afs\athena\user`). (It will also check for custom AFS mount points, and use that name instead of `afs`.)

See <http://www.openafs.org> for more information on OpenAFS.

### 8.7.3 Directories & Subdirectories

A file system is a method of organizing all of the files on an entire disk or hard disk volume. Directories (or folders) are used to divide the files on a disk into logical groups that are easy to work with. Their purpose is similar to that of file drawers containing groups of hanging folders, hanging folders containing smaller folders, and so on. (The terms directory and folder are not synonymous but often used as such in common Windows terminology. For accuracy, we use **directory** throughout these help files unless other folder types are also specifically applicable.)

Every drive has a root or base directory, and many have one or more subdirectories. Subdirectories can also have subdirectories, extending in a branching tree structure from the root directory. The collection of all directories on a drive is often called the directory tree, and a portion of the tree is sometimes called a subtree. The terms directory and subdirectory are typically used interchangeably to mean a single subdirectory within this tree structure.

Subdirectory names follow the same naming rules as files in each operating system (see [File Names](#)).

The drive and subdirectory portions of a file's name are called the file's path. For example, the file name `C:\DIR1\DIR2\MYFILE.DAT` says to look for the file `MYFILE.DAT` in the subdirectory `DIR2` which is part of the subdirectory `DIR1` which is on drive `C`. The path for `MYFILE.DAT` is `C:\DIR1\DIR2`. The backslashes between subdirectory names are required.

Under **TCC**, the path and filename can be up to 32,767 characters, though many Windows applications (including CMD and Explorer) have trouble with path and filename lengths exceeding 260 characters. Shorter paths and names are advisable under Windows whenever feasible.

**TCC** maintains both a current or default drive for your system as a whole, and a current or default directory for every drive in your system. Whenever a program tries to create or access a file without specifying the file's path, the operating system uses the current drive (if no other drive is specified) and the current directory (if no other directory path is specified).

The root directory is named using the drive letter and a single backslash. For example, `D:\` refers to the root directory of drive `D:`. Using a drive letter with no directory name at all refers to the current directory on the specified drive. For example, `E:JPSOFT.DOC` refers to the file `JPSOFT.DOC` in the current directory on drive `E:`, whereas `E:\JPSOFT.DOC` refers to the file `JPSOFT.DOC` in the root directory on drive `E:`.

There are also two special subdirectory names that are useful in many situations: a single period `[.]` means "the current default directory." Two periods `[..]` means "the directory which contains the current default directory" (referred to as the parent directory). These special names can be used wherever a full directory name can be used. **TCC** allows you to use additional periods to specify directories further "up the tree" (see [Extended Parent Directory Names](#)).

Additional information about disk files and file systems is available under [Drives and Volumes](#), [File Systems](#), [File Names](#), and [File Attributes](#).

## 8.7.4 File Names

### FAT File Names

Under the **FAT** file system, a filename consists of a base name of 1 to 8 characters plus an optional extension composed of a period plus 1 to 3 more characters. FAT filenames with an 8-character name and a 3-character extension are sometimes referred to as short filenames (SFNs) to distinguish them from long file names (LFNs).

You can use alphabetic and numeric characters plus the punctuation marks ! # \$ % & ' ( ) - @ ^ \_ ` { } and ~ in both the base name and the extension of a FAT filename. Because the exclamation point [!], percent sign [%], caret [^], at sign [@], parentheses [(]), and back-quote [`] also have other meanings to **TCC**, it is best to avoid using them in filenames. It is also better to use only those characters found in [ASCII](#), because changing font and/or code page may change drastically how they are displayed.

FAT file names are always stored on the disk in upper case, and are displayed in upper or lower case depending on the options you select in **TCC**.

### Long File Names

**VFAT**, **FAT32** and **NTFS** allow using long file names with a maximum of 255 characters, including spaces and other characters that are not allowed in a FAT system file name, but excluding some punctuation characters which are allowed in FAT file names. See your operating system documentation for details on the characters allowed. If you use file names which contain semicolons [;], see [Wildcards](#) for details on avoiding problems with interpretation of those file names under **TCC**.

LFNs are stored and displayed exactly as you entered them, and are not automatically shifted to upper or lower case. For example, you could create a file called *MYFILE*, *myfile*, or *MyFile*, and each name would be stored in the directory just as you entered it. However, case is ignored when looking for filenames, so you cannot have two files whose names differ only in case (*i.e.*, the three names given above would all refer to the same file). This behavior is sometimes described as "case-retentive but not case-sensitive" because the case information is retained, but does not affect access to the files. This is in contrast with Linux-style file systems, which are case sensitive, and permit **AA**, **Aa**, **aA**, and **aa** to be four different file names.

A file that has an LFN may have an additional, "FAT-compatible" name, which contains only those characters legal on a FAT volume, and which meets the 8-character name / 3-character extension limits. Programs which cannot handle long names generally can access files by using their FAT-compatible names. This name is assigned at the time the LFN is created in the specific directory, and to make it unique, it depends on what other SFNs exist in that directory at that instance. Consequently, when copying the file to another directory by its LFN the SFN generated in the target directory may be different from the SFN in the source directory.

When specifying an LFN-compatible file name, which includes spaces or other characters that would either not be allowed in a FAT name, or that may have syntactical significance for **TCC**, you must place double quotes around the name in the command line. For example, suppose you have a file named *LET3* on a FAT volume, and you want to copy it to the *LETTERS* directory on drive F:, an LFN volume, and give it the name *Letter To Sara*. To do so, use either of these commands:

```
copy let3 f:\LETTERS\"Letter To Sara"
copy let3 "f:\LETTERS\Letter To Sara"
```

The LFN file systems do not explicitly define an "extension" for file names which are not FAT-compatible. However, by convention, all characters after the last period in the file name are treated as the extension. For example, the file name "*Letter to Sara*" has no extension, whereas the name "*Letter.to.Sara*" has the extension *Sara*.

Additional information about disk files and file systems is available under [Drives and Volumes](#), [File Systems](#), [Directories and Subdirectories](#), [File Attributes](#), and [Time Stamps](#).

### 8.7.5 File Attributes

Each file has attributes, each of which defines a single characteristic of the file that can be either set or reset. Most file processing commands allow you to select files for processing based on their attributes. The basic attributes Archive, Read only, Hidden, System, and Directory are present on all disk volumes. NTFS volumes support additional attributes: Encrypted, Compressed, Normal, Offline, Temporary, Not content-indexed, Sparse, Junction / Symbolic Link / Reparse point, No Scrub, and Integrity. **Take Command** fully supports these [extended attributes](#).

**Archive** - set by the operating system when the contents of the file are modified to indicate that it is a candidate to be archived, i.e., to be backed up. The attribute can be reset by any program to indicate that the file's contents have been archived. Most programs which can unset this attribute require that you use the explicit reset option, and default to retaining the status of this attribute. For example, the **TCC** command **COPY** requires the **/X** option to reset this attribute.

**Read-only** - if this attribute is set, the file can't be changed or erased accidentally. Most programs honor this attribute by default, which helps to protect important files from erasure and damage.

Either of the **Hidden** and **System** attributes, when set, prevent the file from appearing in directory listings and file searches, including those performed by file processing command of **Take Command**, unless explicitly requested.. This both protects such files from accidental modification, and also speeds up user tasks not explicitly intended to process them.

**Directory** - this attribute is set by the operating system when a subdirectory is created, e.g., by the **MKDIR** command. The attribute cannot be reset. The operating system restricts all accesses to a directory file to directory manipulation operations.

**Volume label** - a special attribute of at most one directory entry in the root directory of a disk drive. The entry can be created, modified, or deleted only through the Windows utility **LABEL** (or equivalent third-party software). **Take Command** does not directly modify the volume label or any of its attributes, and provide read access only through the [VOL](#) command and the [@LABEL\[\]](#) variable function. All other commands ignore this directory entry.

**Normal** - this pseudo attribute is considered to be set if all other attributes (including the [extended attributes](#) available only on an NTFS volumes) are reset. It is not stored by the file system. When **Take Command** checks file attributes, it considers the Normal attribute as set if each of the other attributes is either reset, or unsupported by the combination of the file system and operating system.

The file attributes can also be accessed with the [ATTRIB](#) and [DIR](#) commands, and by the [@ATTRIB](#) and [@WATTRIB](#) variable functions.

Attributes can be set, reset, and viewed with the [ATTRIB](#) command. The [DIR](#) command also has options to view the attribute status of files, and to view information about normally invisible hidden and system files and directories.

## 8.7.6 File Time Stamps

Each file has one or more time stamps. They are used by the operating system to record when the file was created, last modified, or last accessed. Most **TCC** file processing commands allow you to select files for processing based on their time stamps.

1. **Write time** is the date and time the file was last written, i.e., when its content was last modified. On FAT volumes this is the only timestamp. In all commands and functions this is the timestamp used unless you specify another. On FAT and VFAT volumes, the resolution is 2 seconds. NTFS volumes have a 100 nanosecond resolution for the file creation and last write. (UNIX and Linux systems use 1 second resolution.) When a file is copied using the COPY command, even across a network, its write time is not changed. However, different file systems record time with different resolution, so minor changes may occur.
2. **Creation time** is the date and time the current instance of the file was created.
3. **Access time** is the date, and on NTFS volumes, the time, when the file was last accessed for either reading or writing.

Several **TCC** commands and functions let you specify which set of time and date stamps you want to view or work with on LFN volumes. These commands and functions use the letter

- c** creation time stamp,
- w** last write time stamp, and
- a** last access time stamp.

Note that FAT32 and VFAT volumes store the date but not the time of the last access. On these drives the time of last access will always be 00:00.

### **Time Stamp Resolution**

The resolution of time stamps as well as the range of time instances representable vary with file systems.

| <b>file system</b> | <b>resolution</b> | <b>earliest time stamp</b> | <b>latest time stamp</b>  |
|--------------------|-------------------|----------------------------|---------------------------|
| FAT/VFAT           | 2 s               | 1980-01-01 00:00:00 local  | 2107-12-31 23:59:58 local |
| NTFS               | 100 ns            | 1601-01-01 00:00:00 UTC    | 60056-05-28 UTC           |

### **NTFS Timestamp Reports**

These operating systems report timestamps in local time. However, conversion between UTC and local time is based on the difference between UTC and local time at the time of conversion, instead of that in effect when the file event occurred. Consequently, if daylight saving time is currently in effect, all file events around the year will be reported in DST. conversely, when DST is not in effect, all file events around the year will be reported in standard time. This method has the advantage that differences in event times can be calculated easily. However, the times reported will not be those when the event took place if the state DST at time of event is not the same as at the time of reporting.

The [TOUCH](#) command can be used to modify the timestamps of files and directories.

Additional information about disk files and file systems is available under [Drives and Volumes](#), [File Systems](#), [Directories and Subdirectories](#), and [File Names](#).

### 8.7.7 NTFS File Streams

The NTFS file system allows each file to contain multiple "streams" or sets of data. For example a compiler could use streams to store a program's source code, object code, and other data, or a word processing program could use them to store multiple versions of the same document.

Streams are specified by entering a stream name following the file name, for example:

```
myfile.doc:version1
myfile.doc:version2
```

You cannot use wildcards in stream names except when using [filename completion](#).

You can display stream names with the [DIR /:](#) option. The file processing commands [COPY](#), [DEL](#), [FFIND](#), [HEAD](#), [LIST](#), [MOVE](#), [TAIL](#) and [TYPE](#) support file streams when the stream name is explicitly specified; see the individual commands for additional details. Other file-related commands, such as ATTRIB and TOUCH work with the file as a whole, and not with any particular stream or portion of the file data.

Variable functions which reference file contents, such as [@FILEOPEN](#), [@LINE](#), and [@LINES](#) also accept stream names.

## 8.8 Regular Expression Syntax

**Oniguruma Regular Expressions** Version 5.9.2 2010/01/09

This section covers the Ruby regular expression syntax. For information on Perl regular expression syntax, see your Perl documentation or <http://www.perl.com/doc/manual/html/pod/perlre.html>.

### 1. Syntax elements

|       |                                                   |
|-------|---------------------------------------------------|
| \     | escape (enable or disable meta character meaning) |
|       | alternation                                       |
| (...) | group                                             |
| [...] | character class                                   |

### 2. Characters

|              |                                                    |
|--------------|----------------------------------------------------|
| \t           | horizontal tab (0x09)                              |
| \v           | vertical tab (0x0B)                                |
| \n           | newline (0x0A)                                     |
| \r           | return (0x0D)                                      |
| \b           | back space (0x08)                                  |
| \f           | form feed (0x0C)                                   |
| \a           | bell (0x07)                                        |
| \e           | escape (0x1B)                                      |
| \nnn         | octal char (encoded byte value)                    |
| \xHH         | hexadecimal char (encoded byte value)              |
| \x{7HHHHHHH} | wide hexadecimal char (character code point value) |
| \cx          | control char (character code point value)          |

\C-x control char (character code point value)  
 \M-x meta (x|0x80) (character code point value)  
 \M-\C-x meta control char (character code point value)

(\* \b is effective in character class [...] only)

### 3. Character types

. any character (except newline)

\w word character

Not Unicode:  
 alphanumeric, "\_" and multibyte char.

Unicode:  
 General\_Category -- (Letter|Mark|Number|Connector\_Punctuation)

\W non word char

\s whitespace char

Not Unicode:  
 \t, \n, \v, \f, \r, \x20

Unicode:  
 0009, 000A, 000B, 000C, 000D, 0085(NEL),  
 General\_Category -- Line\_Separator  
 -- Paragraph\_Separator  
 -- Space\_Separator

\S non whitespace char

\d decimal digit char

Unicode: General\_Category -- Decimal\_Number

\D non decimal digit char

\h hexadecimal digit char [0-9a-fA-F]

\H non hexadecimal digit char

Character Property

\* \p{property-name}  
 \* \p{^property-name} (negative)  
 \* \P{property-name} (negative)

property-name:

+ works on all encodings

Alnum, Alpha, Blank, Cntrl, Digit, Graph, Lower, Print, Punct, Space, Upper, XDigit,  
Word, ASCII,

#### 4. Quantifier

greedy

|                                                       |
|-------------------------------------------------------|
| ?      1 or 0 times                                   |
| *      0 or more times                                |
| +      1 or more times                                |
| {n,m}    at least n but not more than m times         |
| {n,}     at least n times                             |
| {,n}     at least 0 but not more than n times ({0,n}) |
| {n}      n times                                      |

reluctant

|                                                            |
|------------------------------------------------------------|
| ??     1 or 0 times                                        |
| *?     0 or more times                                     |
| +?     1 or more times                                     |
| {n,m}?    at least n but not more than m times             |
| {n,}?     at least n times                                 |
| {,n}?     at least 0 but not more than n times (== {0,n}?) |

possessive (greedy and does not backtrack after repeated)

|                        |
|------------------------|
| ?+     1 or 0 times    |
| *+     0 or more times |
| ++     1 or more times |

({n,m}+, {n,}+, {n}+ are possessive op. in ONIG\_SYNTAX\_JAVA only)

ex. /a\*+/ === /(>?a\*)/

#### 5. Anchors

|    |                                             |
|----|---------------------------------------------|
| ^  | beginning of the line                       |
| \$ | end of the line                             |
| \b | word boundary                               |
| \B | not word boundary                           |
| \A | beginning of string                         |
| \Z | end of string, or before newline at the end |
| \z | end of string                               |
| \G | matching start position (*)                 |

#### 6. Character class

|       |                                             |
|-------|---------------------------------------------|
| ^...  | negative class (lowest precedence operator) |
| x-y   | range from x to y                           |
| [...] | set (character class in character class)    |

`..&&..` intersection (low precedence at the next of ^)

ex. `[a-w&&[^c-g]z] ==> ([a-w] AND ([^c-g] OR z)) ==> [abh-w]`

\* If you want to use '[', '^', ']' as a normal character in a character class, you should escape these characters by '\'.

POSIX bracket (`([:xxxxx:], negate [:^xxxxx:])`)

#### Not Unicode Case:

|          |                                             |
|----------|---------------------------------------------|
| alnum    | alphabet or digit char                      |
| alpha    | alphabet                                    |
| ascii    | code value: [0 - 127]                       |
| blank    | \t, \x20                                    |
| cntrl    |                                             |
| digit0-9 |                                             |
| graph    | include all of multibyte encoded characters |
| lower    |                                             |
| print    | include all of multibyte encoded characters |
| punct    |                                             |
| space    | \t, \n, \v, \f, \r, \x20                    |
| upper    |                                             |
| word     | alphanumeric, "_" and multibyte characters  |
| xdigit   | 0-9, a-f, A-F                               |

#### Unicode Case:

|        |                                                                                                                                               |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| alnum  | Letter   Mark   Decimal_Number                                                                                                                |
| alpha  | Letter   Mark                                                                                                                                 |
| ascii  | 0000 - 007F                                                                                                                                   |
| blank  | Space_Separator   0009                                                                                                                        |
| cntrl  | Control   Format   Unassigned   Private_Use   Surrogate                                                                                       |
| digit  | Decimal_Number                                                                                                                                |
| graph  | [:^space:] && ^Control && ^Unassigned && ^Surrogate                                                                                           |
| lower  | Lowercase_Letter                                                                                                                              |
| print  | [:graph:]   [:space:]                                                                                                                         |
| punct  | Connector_Punctuation   Dash_Punctuation   Close_Punctuation   Final_Punctuation   Initial_Punctuation   Other_Punctuation   Open_Punctuation |
| space  | Space_Separator   Line_Separator   Paragraph_Separator   0009   000A   000B   000C   000D   0085                                              |
| upper  | Uppercase_Letter                                                                                                                              |
| word   | Letter   Mark   Decimal_Number   Connector_Punctuation                                                                                        |
| xdigit | 0030 - 0039   0041 - 0046   0061 - 0066 (0-9, a-f, A-F)                                                                                       |

## 7. Extended groups

|            |                                                                         |
|------------|-------------------------------------------------------------------------|
| (?#...)    | comment                                                                 |
| (?imx-imx) | option on/off<br>i: ignore case<br>m: multi-line (dot(.) match newline) |

(?imx-imx:subexp) x: extended form  
option on/off for subexp

(?:subexp) not captured group  
(subexp) captured group

(?=subexp) look-ahead  
(?!subexp) negative look-ahead  
(?<=subexp) look-behind  
(?<!subexp) negative look-behind

Subexp of look-behind must be fixed character length. But different character length is allowed in top level alternatives only.  
ex. (?<=a|bc) is OK. (?<=aaa(?:b|cd)) is not allowed.

In negative-look-behind, captured group isn't allowed, but shy group(?:) is allowed.

(?>subexp) atomic group  
don't backtrack in subexp.

(?<name>subexp) define named group  
(All characters of the name must be a word character. And first character must not be a digit or upper case)  
Not only a name but a number is assigned like a captured group.

Assigning the same name as two or more subexps is allowed. In this case, a subexp call can not be performed although the back reference is possible.

## 8. Back reference

|          |                                                  |
|----------|--------------------------------------------------|
| \n       | back reference by group number (n >= 1)          |
| \k<n>    | back reference by group number (n >= 1)          |
| \k'n'    | back reference by group number (n >= 1)          |
| \k<-n>   | back reference by relative group number (n >= 1) |
| \k'-n'   | back reference by relative group number (n >= 1) |
| \k<name> | back reference by group name                     |
| \k'name' | back reference by group name                     |

In the back reference by the multiplex definition name, a subexp with a large number is referred to preferentially. (When not matched, a group of the small number is referred to.)

\* Back reference by group number is forbidden if named group is defined in the pattern and ONIG\_OPTION\_CAPTURE\_GROUP is not setted.

Back reference with nest level

level: 0, 1, 2, ...

\k<n+level> (n >= 1)  
\k<n-level> (n >= 1)  
\k'n+level' (n >= 1)

\k'n-level' (n >= 1)

\k<name+level>  
\k<name-level>  
\k'name+level'  
\k'name-level'

Destinate relative nest level from back reference position.

example 1.

```
\A(?<a>|.|\(?:(?.)\g<a>\k<b+0>))\z/.match("reer")
```

example 2.

```
r = Regexp.compile(<<'_REGEXP_'.strip, Regexp::EXTENDED)
(?<element> \g<stag> \g<content>* \g<etag>){0}
(?<stag> < \g<name> \s* >){0}
(?<name> [a-zA-Z_-]+){0}
(?<content> [^<&]+ (\g<element> | [^<&]+)*){0}
(?<etag> </ \k<name+1> >){0}
\g<element>
REGEXP
```

```
p r.match('<foo>f<bar>bbb</bar>f</foo>').captures
```

## 9. Subexp call ("Tanaka Akira special")

|          |                                        |
|----------|----------------------------------------|
| \g<name> | call by group name                     |
| \g'name' | call by group name                     |
| \g<n>    | call by group number (n >= 1)          |
| \g'n'    | call by group number (n >= 1)          |
| \g<-n>   | call by relative group number (n >= 1) |
| \g'-n'   | call by relative group number (n >= 1) |

\* left-most recursive call is not allowed.

ex. (?<name>a|\g<name>b) => error  
      (?<name>a|b|\g<name>c) => OK

\* Call by group number is forbidden if named group is defined in the pattern and ONIG\_OPTION\_CAPTURE\_GROUP is not set.

\* If the option status of called group is different from calling position then the group's option is effective.

ex. (?-i:\g<name>)(?i:(?<name>a)){0} match to "A"

## 10. Captured group

Behavior of the no-named group (...) changes with the following conditions. (But named group is not

changed.)

case 1. `./...` (named group is not used, no option)

(...) is treated as a captured group.

case 2. `./..g` (named group is not used, 'g' option)

(...) is treated as a no-captured group (?...).

case 3. `./..(?<name>..)../` (named group is used, no option)

(...) is treated as a no-captured group (?...).

numbered-backref/call is not allowed.

case 4. `./..(?<name>..)/G` (named group is used, 'G' option)

(...) is treated as a captured group.

numbered-backref/call is allowed.

where

`g`: `ONIG_OPTION_DONT_CAPTURE_GROUP`

`G`: `ONIG_OPTION_CAPTURE_GROUP`

### A-1. Syntax dependent options

+ RUBY

(?m): dot(.) match newline

+ PERL and JAVA

(?s): dot(.) match newline

(?m): ^ match after newline, \$ match before newline

### A-2. Original extensions

+ hexadecimal digit char type `\h`, `\H`

+ named group `(?<name>...)`

+ named backref `\k<name>`

+ subexp call `\g<name>, \g<group-num>`

### A-3. Missing features compared with Perl 5.8.0

+ `\N{name}`

+ `\I, \u, \L, \U, \X, \C`

+ `(?{code})`

+ `(??{code})`

+ `(?(condition)yes-pat|no-pat)`

\* `\Q...\E`

This is effective in PERL and JAVA.

## 8.9 XML in TCC

**TCC** provides the ability to open, parse and close XML documents through the use of a subset of the XPath language. The syntax of the language is explained more clearly at the W3C site: [http://www.w3schools.com/xpath/xpath\\_syntax.asp](http://www.w3schools.com/xpath/xpath_syntax.asp). We support a limited subset of the language explained below. Any functions of the language beyond what are listed below are unsupported, but may work.

The most common use of XML in **TCC** is to parse an XML formatted data file to extract elements for further processing in Take Command.

There are four **TCC** functions that provide XML support:

1. **@XMLOPEN** - open an XML file for use by @XMLPATH and/or @XMLNODES. The syntax is:

`@XMLOPEN[filename]`

Example: set a=%@XMLOPEN[bookstore.xml]

2. **@XMLCLOSE** - close an XML file previously opened by @XMLOPEN. The syntax is:

`@XMLCLOSE[]`

Example: set a=%@XMLCLOSE[]

3. **@XMLNODES** - return the number of nodes (children) for the specified path in an XML file. The syntax is:

`@XMLNODES[["filename"],path]`

Example: set a=%@XMLNODES["bookstore.xml",/bookstore] - returns the number of books in the bookstore file (see example file below)

If you don't specify a filename (which \*must\* be in double quotes), @XMLNODES will use the XML file previously opened by @XMLOPEN.

4. **@XMLPATH** - XML XPath query. (See the XML XPath docs for details on XPath syntax.) The syntax is:

`@XMLPATH[["filename"],path]`

Example: echo %@XMLPATH[/bookstore/book] - lists all the sub-element values for the first book (see example file below)

If you don't specify a filename (which \*must\* be in double quotes), @XMLPATH will use the XML file previously opened by [@XMLOPEN](#).

To return an attribute, preface the attribute name with an @.

### Typical Use of The XML Functions

All discussions in this section refer to the following XML data file.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<bookstore>
```

```

<book>
 <title lang="jap">Harry Potter</title>
 <price>29.99</price>
</book>
<book>
 <title lang="eng">Learning XML</title>
 <price>39.95</price>
</book>
<book>
 <title lang="ger">Day Watch</title>
 <price>14.99</price>
</book>
<book>
 <title lang="eng">Winston Churchill: An Autobiography</title>
 <price>49.99</price>
</book>
</bookstore>

```

Each tag in this file is referred to as a node (e.g., bookstore, book, title). The identifiers within a tag are often referred to as an attribute (e.g., lang).

Typically you will use the four commands in the following order:

1. Open the file with @XMLOPEN
2. Get the number of child nodes (records) to process with @XMLNODES
3. Set up a loop from one to the number of records to process the records with @XMLXPATH
4. Close the file with @XMLCLOSE

Below is a simple example of batch processing (without error handling J):

```

SET a=%@XMLOPEN[bookstore.xml]
SET b=%@XMLNODES[/bookstore]
DO i = 1 to %b
 SET Title= %@XMLXPATH[/bookstore/book[%i]/title]
 SET Price= %@XMLXPATH[/bookstore/book[%i]/price]
 ECHO %Title ` costs only ` %Price
ENDDO
SET c=%@XMLCLOSE[]

```

This example:

- Opens the bookstore data file,
- Evaluates the number of books in the bookstore as 4(since the only child of bookstore is books),
- Gets the title and price for each book
- Prints each title and price out
- Closes the data file.

XPath offers a large number of processing options in addition to the ones above. We have not tested their syntax. If you wish to test additional functionality and report back to us, we will add that information to our documentation.

## 8.10 Miscellaneous

- [Executable Files and File Searches](#)
- [Popup Windows](#)
- [Windows System Errors](#)
- [ASCII Codes and Key Names](#)
- [ANSI X3.64 Command Reference](#)
- [Colors, Colors Names & Codes](#)

### 8.10.1 Executable Files & File Searches

When **TCC** can't find a matching internal command name, it tries to find an executable file whose name matches the command name. (Executable files are typically those with an *.EXE* extension.)

If **TCC** cannot find an executable program to run, it next looks for a matching [batch file](#) name. **TCC** looks first for a *.BTM* file, then for a *.CMD* file, then for a *.BAT* file, and finally for a *.REX*, *.REXX*, *.PL*, *.PY*, *.RB*, or *.TCL* file (if REXX, Perl, Python, Ruby, and/or Tcl are enabled).

You can change the list of extensions that are considered "executable", and the order in which they are searched, with the [PATHEXT](#) environment variable, and the related [PathExt](#) configuration option. PATHEXT is supported for compatibility reasons but should not generally be used as a substitute for [executable extensions](#), which are more flexible.

**Note:** If the search for an external program or batch file fails, **TCC** checks to see if the command name matches the name of a file with an [executable extension](#). If an executable extension is found, **TCC** runs the program specified when the association was defined. If no executable extension is found, **TCC** will look for a direct association for the extension in the registry and insert the associated string (usually the name of an application) at the beginning of the command line, then call the Windows CreateProcess API to execute that command. If the CreateProcess call fails, or if no association was found in the registry, **TCC** calls the ShellExec Windows API. **TCC** has no control over which action the above Windows APIs will take when presented with a file name. If you are concerned about what Windows might do with an "unknown" extension, create a specific executable extension.

**TCC** first performs this search (for an executable program, a batch file, or a file with an executable extension) in the current directory. If that search fails, they repeat the search in every directory in your search path.

The search path is a list of directories that **TCC** (and some applications) search for executable files. For example, if you wanted **TCC** to search the root directory of the C: drive, the \WINUTIL subdirectory on the C: drive, and the \UTIL directory on the D: drive for executable files, your search path would look like this:

```
PATH=C:\;C:\WINUTIL;D:\UTIL
```

The directory names in the search path are separated by semicolons.

You can create or view the search path with the [PATH](#) command. You can use the [ESET](#) command to edit the path. Many programs also use the search path to find their own files. The search path is stored in the environment with the name PATH.

**Take Command** also searches the \WINDOWS\SYSTEM32 directory followed by the \WINDOWS directory. (The actual directory names may be different on your system. **TCC** will determine the correct

names for the "Windows" and "Windows System" directories and use them.) This part of the search procedure conforms with the traditional search sequences used under each Windows operating system.

**Note:** If the file is not found on the PATH, **TCC** then checks for a corresponding **App Paths** entry in the Windows registry. **App Paths** entries are created by some applications during the installation process.

Remember, **TCC** always looks for an executable file (or a file with an executable extension or Windows file association) in the current subdirectory, then in the Windows directories if appropriate (see above), then in each directory in the search path, and then in the **App Paths** area of the registry. (You can change the search order so the current directory is not searched first; see the [PATH](#) command for details.)

If you include an extension as part of the command name, **TCC** only searches for a file with that extension. Similarly, if you include a path as part of the command name, **TCC** will look only in the directory you specified, and ignore the usual search of the current directory and the PATH.

If your command name includes a path, the elements must be separated with backslashes (e.g. `c:\wp\wp`). If you are accustomed to Linux syntax where forward slashes are used in command paths, and want **TCC** to recognize this approach, you can set the [Unix/Linux Paths](#) configuration option.

Once the file is found, **TCC** executes it based on its extension. `.EXE` files are executed by passing their names to the operating system. `.BTM`, `.BAT`, and (if applicable) `.CMD` files are executed by **TCC**, which reads each line in the file as a new command. Files with executable extensions are executed by starting the associated application, and passing the name of the file on the command line.

If you specify a file name including extension, and the file exists in the current directory (or you specify a path), but the file does not have an extension known to **TCC** (`.EXE`, `.BTM`, `.BAT`, `.CMD`, or an executable extension), then the file name will be passed to Windows to check for file associations defined in the Windows registry. This allows you to execute any file whose extension is known to Windows, simply by typing its name. For example, if you have no executable extension defined for `.PSP` files, but this is an extension known to Windows, at the prompt you can simply enter a command like this:

```
[c:\graphics] image1.psp
```

and **Take Command** will request that Windows start the application for you. See [Windows File Associations](#) for additional details on how to control Windows file associations in **TCC**.

The following table sums up the possible search options (the term "standard search" refers to the search of the current directory, the Windows directories, and each directory in the search path):

<b>Command</b>	<b>TCC Search Sequence</b>
WP	Search for any executable file whose base name is <i>WP</i> .
WP.EXE	Search for <code>WP.EXE</code> ; will not find files with other extensions.
C:\WP\WP	Looks in the <code>C:\WP\WP</code> directory for any executable file whose base name is <i>WP</i> . Does not check the standard search directories.
C:\WP\WP.EXE	Looks only for the file <code>C:\WP\WP.EXE</code> .
LAB.DOC	Search for <code>LAB.DOC</code> , if <code>.DOC</code> is defined as an executable extension. Runs the associated application if the file is found. If <code>.DOC</code> is not an executable extension, passes the name to Windows to check for a Windows file association.
C:\L\LAB.DOC	Looks only for the file <code>C:\L\LAB.DOC</code> , and only if <code>.DOC</code> is defined as an executable extension. Runs the associated application if the file is found. If <code>.DOC</code> is not an executable extension, passes the name to Windows to check

for a Windows file association.

If **TCC** cannot find an executable file, batch program, or a file with an executable extension or Windows file association in the current directory, a directory in the search path, or the directory you specified in the command, it then looks for an alias called **UNKNOWN\_CMD** (see the [ALIAS](#) command for details). If you have defined an alias with that name, it is executed (this allows you to control error handling for unknown commands). If **TCC** cannot find an **UNKNOWN\_CMD** alias, it will look for a plugin command named **UNKNOWN\_CMD**. Otherwise, **TCC** displays an "Unknown command" error message and waits for your next instruction.

See also: the [WHICH](#) command.

#### 8.10.1.1 Windows File Associations

Windows includes the ability to associate file extensions with specific applications. For example, a graphics program might be associated with files with a **.JPG** extension, while Notepad could be associated with files with a **.TXT** extension.

When you attempt to start an application from the command line or a batch file, **TCC** first searches for an external program file with a standard extension (**.EXE**, **.CMD**, etc.). It then checks executable extensions. If all of these tests fail, **TCC** passes the command name to Windows to see if Windows can find an association for it.

**TCC** offers two commands which provide control over file associations. Both should be used with caution to avoid creating errors in the registry or damaging existing file types. The [ASSOC](#) command modifies or displays the associations between extensions and file types in the Windows registry. The [FTYPE](#) command modifies or displays the default command used to "open" a file of a specified type.

Executable extensions defined in **TCC** always take precedence over file associations defined in Windows. For example, if you associate the **.TXT** extension with your own editor using a **TCC** executable extension, and Windows has associated **.TXT** with Notepad, your setting will have priority, and the association with Notepad will be ignored when you invoke a **.TXT** file from within **TCC**.

See also: [START](#), [ASSOC](#), [FTYPE](#), [Executable Extensions](#), [Executable Files and File Searches](#).

#### 8.10.2 Popup Windows

Several features of **TCC** display popup windows. A popup window may be used to display filenames, recently-executed commands, recently-used directories, the results of an [extended directory search](#), or a list created by the [SELECT](#) command or the [@SELECT](#) internal function.

Popup windows always display a list of choices and a cursor bar. You can move the cursor bar inside the window until you find the choice that you wish to make, then press the **Enter** key to select that item.

Navigation inside any popup window follows the conventions described below. Additional information on each specific type of popup window is provided where that window is discussed in detail.

The popup windows can be moved and resized, and will remember their position and size when recalled. (Not supported in **TCC/LE**.) You can also change the keys used in popup windows with [key mapping directives](#).

Once a window is open, you can use these navigation keys to find the selection you wish to make:

Up Arrow

Move the selection bar up one line

Down Arrow	Move the selection bar down one line
Left Arrow	Scroll the display left 1 column, if it is a scrolling display (i.e. if it has a horizontal scrollbar)
Right Arrow	Scroll the display right 1 column, if it is a scrolling display (i.e. if it has a horizontal scrollbar)
PgUp	Scroll the display up one page
PgDn	Scroll the display down one page
Home	Go to the beginning of the list
End	Go to the end of the list
<a href="#"><u>Esc</u></a>	Close the window without making a selection
<a href="#"><u>Enter</u></a>	Select the current item and close the window
<a href="#"><u>Ctrl-E</u></a>	Edit the current selection
<a href="#"><u>Ctrl-D</u></a>	Delete the current selection

**Note:** The keystrokes shown above are the defaults values. See [Key Mapping Directives](#) for details on how to assign different keystrokes.

All of the popup windows have an edit control on the toolbar. Entering a search string there (or just typing while the popup window has focus) will eliminate non-matching entries from the window. The search string can also contain wildcards or regular expressions. If the string doesn't contain any wildcards or a leading :: (for a regular expression), **TCC** will append a \* to the string (to match any line beginning with the string). For example, entering \*jisoft in the edit control at the top of the window will select all matching lines that contain "jisoft" anywhere.

The **TCC** popup windows can optionally use character-mode windows instead of GUI windows. This is intended for use with server consoles that are character-mode only, or when using SSH with no GUI support; there is no benefit (and several disadvantages) in using this option for normal non-server environments. See the TCMD.INI [ConsolePopupWindows](#) directive.

### 8.10.3 Windows System Errors

System errors are internal errors returned by Windows when Windows APIs fail. You can retrieve the number of the most recent Windows system error with the internal variable [SYSERR](#).

For a detailed list of Windows system error codes, see:

[http://msdn2.microsoft.com/en-us/library/ms681381\(VS.85\).aspx](http://msdn2.microsoft.com/en-us/library/ms681381(VS.85).aspx)

### 8.10.4 ASCII Codes and Key Names

For ASCII codes and key names see:

- [ASCII Tables](#)
- [Keys & Key Names](#)

The remainder of this section gives a explanation of the ASCII character sets and key names. For more information on **TCC**'s ANSI X3.64 string support see [ANSI X3.64 Commands Reference](#). If you are troubleshooting a keyboard or character display problem, be sure to read all of the explanation below before referring to the tables.

The translation of a key you type on the keyboard to a displayed character on the screen depends on several related aspects of character handling. A complete discussion of these topics is well beyond the

scope of this document. However, a basic picture of the steps in the keystroke and character translation process will help you understand how characters are processed in your system, and why they occasionally may not come out the way you expect.

Internally, computers use numbers to represent the keys you press and the characters displayed on the screen. To display the text that you type, your computer and operating system require five pieces of information:

1. The numeric key code for the physical key you pressed (determined by your keyboard hardware);
2. The specific character that key code represents based on your current keyboard layout or country setting;
3. The character set currently in use on your system (see below);
4. The international code page in use for that character set; and
5. The display font used to display the character.

If the key codes produced by your keyboard, the code page, and the font you choose are not fully compatible, the characters displayed on the screen will not match what you type. The differences are likely to appear in line-drawing characters, "international" (non-English) characters, and special symbols, but not in commonly-used U.S. English alphabetic, numeric, or punctuation characters.

The control codes can be entered on most keyboards by pressing the **Ctrl** key plus another character, or by pressing the special keys **Tab**, **Enter**, **Backspace**, and **Esc**.

See your operating system documentation for more information about character sets, code pages, and country and language support. Refer to your operating system and/or font documentation for details on the full character set available in any particular font.

The tables in this section are based on U.S. English conventions. Your system may differ if it is configured for a different country or language. See your operating system documentation for more information about country and language support.

**Note:** You may also be able to use the **Alt + keypad** approach to generate ASCII values. See "[Command Line Editing](#)" for additional information.

#### 8.10.4.1 ASCII Tables

These tables show the 128-character ASCII set for U.S. English systems. Most of the characters in code range 32..126 (the only codes for which ASCII specifies displayable symbols) will be the same on non-U.S. systems. The symbols associated with all other codes vary from font to font, as well as from country to country.

For more details on ASCII, character sets, and key codes, see the general information topic on [ASCII, Key Codes, and ANSI X3.64 Commands](#).

- [Control Characters 0 - 31, 127](#)
- [Printing Characters 32 - 47](#)
- [Printing Characters 48 - 63](#)
- [Printing Characters 64 - 79](#)
- [Printing Characters 80 - 95](#)
- [Printing Characters 96 - 111](#)
- [Printing Characters 112 - 126](#)

**Control Characters 0 - 31, 127**

<b>ASCII (Dec)</b>	<b>ASCII (Hex)</b>	<b>Ctrl + Key</b>	<b>Acronym</b>	<b>Name</b>
0	00	@	NUL	null
1	01	A	SOH	start of header
2	02	B	STX	start text
3	03	C	ETX	end text
4	04	D	EOT	end of transmission
5	05	E	ENQ	enquiry
6	06	F	ACK	acknowledge
7	07	G	BEL	bell
8	08	H	BS	backspace
9	09	I	HT	horizontal tab
10	0A	J	LF	linefeed
11	0B	K	VT	vertical tab
12	0C	L	FF	form feed
13	0D	M	CR	carriage return
14	0E	N	SO	shift out
15	0F	O	SI	shift in
16	10	P	DLE	data link escape
17	11	Q	DC1	device control 1
18	12	R	DC2	device control 2
19	13	S	DC3	device control 3
20	14	T	DC4	device control 4
21	15	U	NAK	negative acknowledge
22	16	V	SYN	synchronize
23	17	W	ETB	end text block
24	18	X	CAN	cancel
25	19	Y	EM	end of medium
26	1A	Z	SUB	substitute
27	1B	[	ESC	escape
28	1C	\	FS	field separator
29	1D	]	GR	group separator
30	1E	^	RS	record separator
31	1F	_	US	unit separator
127	7F	n/a	DEL	delete

**Printing Characters 32 - 47**

<b>Dec</b>	<b>Hex</b>	<b>Char</b>	<b>Special character name</b>
032	20	Space	space
033	21	!	exclamation mark
034	22	"	quote mark
035	23	#	number sign

036	24	\$	dollar (currency) sign
037	25	%	percent mark
038	26	&	ampersand
039	27	'	apostrophe
040	28	(	left parenthesis
041	29	)	right parenthesis
042	2A	*	asterisk
043	2B	+	plus sign
044	2C	,	comma
045	2D	-	hyphen (minus sign)
046	2E	.	period
047	2F	/	slash

**Printing Characters 48 - 63**

<b>Dec</b>	<b>Hex</b>	<b>Char</b>	<b>Special character name</b>
048	30	0	
049	31	1	
050	32	2	
051	33	3	
052	34	4	
053	35	5	
054	36	6	
055	37	7	
056	38	8	
057	39	9	
058	3A	:	colon
059	3B	;	semicolon
060	3C	<	less than sign
061	3D	=	equal sign
062	3E	>	greater than sign
063	3F	?	question mark

**Printing Characters 64 - 79**

<b>Dec</b>	<b>Hex</b>	<b>Char</b>	<b>Special character name</b>
064	40	@	at sign
065	41	A	
066	42	B	
067	43	C	
068	44	D	
069	45	E	
070	46	F	
071	47	G	
072	48	H	
073	49	I	
074	4A	J	

075	4B	K	
076	4C	L	
077	4D	M	
078	4E	N	
079	4F	O	

**Printing Characters 80 - 95**

<b>Dec</b>	<b>Hex</b>	<b>Char</b>	<b>Special character name</b>
080	50	P	
081	51	Q	
082	52	R	
083	53	S	
084	54	T	
085	55	U	
086	56	V	
087	57	W	
088	58	X	
089	59	Y	
090	5A	Z	
091	5B	[	left bracket
092	5C	\	backslash
093	5D	]	right bracket
094	5E	^	caret
095	5F	_	underscore

**Printing Characters 96 - 111**

<b>Dec</b>	<b>Hex</b>	<b>Char</b>	<b>Special character name</b>
096	60	`	accent grave (back tick or back quote)
097	61	a	
098	62	b	
099	63	c	
100	64	d	
101	65	e	
102	66	f	
103	67	g	
104	68	h	
105	69	i	
106	6A	j	
106	6B	k	
108	6C	l	
109	6D	m	
110	6E	n	
111	6F	o	

### Printing Characters 112 - 126

<u>Dec</u>	<u>Hex</u>	<u>Char</u>	<u>Special character name</u>
112	70	p	
113	71	q	
114	72	r	
115	73	s	
116	74	t	
117	75	u	
118	76	v	
119	77	w	
120	78	x	
121	79	y	
122	7A	z	
123	7B	{	left brace
124	7C		vertical bar
125	7D	}	right brace
126	7E	~	tilde

#### 8.10.4.2 Key Names

Key names are used by **Take Command** in [tab toolbar](#) buttons, and in **TCC** to define keystroke [aliases](#), in [key mapping directives](#), and in the [INKEY](#) and [KEYSTACK](#) commands. The format of a key name is the same in all four cases:

[Prefix-]Keyname

The valid prefix and keyname combinations are shown in the table below. Names of keys must be spelled exactly as shown, except for case. Note that you cannot specify a punctuation key.

<b>Prefix</b>	<b>Valid for keynames</b>
none	A-Z, 0-9, F1-F12, Tab, Bksp, Enter, Up, Down, Left, Right, PgUp, PgDn, Home, End, Ins, Del, Esc
Alt-	A-Z, 0-9, F1-F12, Bksp, and the non-alphanumeric keys `-=[]\';./
Ctrl-	A-Z, F1-F12, Tab, Bksp, Enter, Up, Down, Left, Right, PgUp, PgDn, Home, End, Ins, Del
Shift-	F1-F12, Tab

The prefix and key name must be separated by a hyphen (-). For example:

Alt-F10 ctrl-bksp

Some keys are intercepted by Windows and are not passed on to **Take Command**. For example, **Alt-Tab**, **Alt-Esc** and **Ctrl-Esc** typically pop up a task list, or are used in switching among multiple tasks. **Alt-space** brings down a menu to control window size and position, etc. Keys which are intercepted by the operating system (including menu accelerators, i.e. **Alt** plus another key) generally cannot be assigned to [aliases](#) or with [key mapping directives](#), because **Take Command** never receives these keystrokes. However, [KEYSTACK](#) can send them to Windows (though not to another application).

The above comments are based on common 101/102-key US-style keyboards. Some key combinations might not be available on some keyboards.

### 8.10.5 ANSI X3.64 Command Reference

**TCC** includes support for ANSI Std X3.64, allowing you to manipulate the cursor, screen color, and other display attributes through sequences of special characters embedded in the text sent to the display. These sequences are called "ANSI commands". (For a general description of this feature, see [ANSI Support](#).) Because of the design of Windows, **TCC** cannot provide ANSI X3.64 support to external applications.

**TCC** supports most common ANSI X3.64 screen commands, but does not provide the complete set of options supported by some operating system's ANSI X3.64 drivers (for example, **TCC** does not include ANSI X3.64 key substitutions; that functionality is already provided with [key aliases](#)). This section is a quick reference to the ANSI X3.64 commands supported by **TCC**.

ANSI X3.64 support within **TCC** can be enabled or disabled with the [ANSI Colors](#) configuration option, or the [SETDOS](#) /A command. You can test whether ANSI X3.64 support is enabled with the [ANSI](#) internal variable.

An ANSI X3.64 command string consists of three parts:

- |                     |                                                                                                                           |
|---------------------|---------------------------------------------------------------------------------------------------------------------------|
| <b>&lt;ESC&gt;[</b> | The ASCII character ESC, followed by a left bracket. These two characters must be present in all ANSI X3.64 strings.      |
| <b>parameters</b>   | Optional parameters for the command, usually numeric. If there are multiple parameters, they are separated by semicolons. |
| <b>command</b>      | A single-letter command. (Case sensitive!)                                                                                |

For example, to position the cursor to row 7, column 12 the ANSI X3.64 command is:

<ESC>[7;12H

The **parameters** part of this command is "7;12" and the **command** part is "H".

To transmit ANSI X3.64 commands to the screen you can use the [ECHO](#) command. The ESC character can be generated by inserting it into the string directly (if you are putting the string in a batch file and your editor will insert such a character), or by using the internal "[escape" character](#) (defaults: caret, [^]) followed by a lower-case "e".

For example, the sequence shown above could be transmitted from a batch file with either of these commands (the first uses an ESC character directly, represented below by "<ESC>"; the second uses ^e):

```
echo <ESC>[7;12H
echo ^e[7;12H
```

You can also include ANSI X3.64 commands in your [prompt](#), using \$e to send the <ESC> character.

#### Commands

The internal **TCC** ANSI X3.64 interpreter supports the subset of X3.64 commands below. Variable parameters are shown in lower-case italics, e.g., **row** and **attr**, and must be replaced with the appropriate decimal numeric value when using the commands. The default value for **row**, **rows**, **col**, and **cols** is 1.

<ESC>[rowsA	Cursor up by <i>rows</i>
<ESC>[rowsB	Cursor down by <i>rows</i>
<ESC>[colsC	Cursor right by <i>cols</i>
<ESC>[colsD	Cursor left by <i>cols</i>
<ESC>[row;colH	Set cursor position (top left is row 1, column 1)
<ESC>[2J	Clear whole screen
<ESC>[K	Clear from cursor to end of line
<ESC>[row;colf	Set cursor position, same as "H" command
<ESC>[attr1;attr2;...m	Set display attributes; see table of attribute values below
<ESC>[s	Save cursor position (may not be nested)
<ESC>[u	Restore saved cursor position

## Display Attributes

The attribute values used for the **m** command are:

- 0** Restore all attributes to default
- 1** Bright (high intensity) foreground color
- 2** Normal intensity foreground color
- 5** Bright (high intensity) background
- 7** Reverse video
- 30..37** Foreground color
- 40..47** Background color

Foreground Code	Background Code	Color
30	40	Black
31	41	Red
32	42	Green
33	43	Yellow
34	44	Blue
35	45	Magenta
36	46	Cyan
37	47	White

If you are setting multiple attributes, combine them into a single command (using the ; concatenation operator). The attribute settings are cumulative, and are independent of order (except code **0**, reset to default).

## Examples

Set bright red foreground without changing background:

```
echo ^e[31;1m
```

Set the display to bright cyan on blue, and clear the screen:

```
echo ^e[44;36;1m^e[2J
```

Set up a prompt which saves the cursor position, displays the date and time on the top line in bright white on magenta, and then restores the cursor position and sets the color to bright cyan on blue, and displays the standard prompt:

```
prompt $e[s$e[1;1f$e[45;37;1m$e[K$d te[u$e[44;36;1mpg
```

## 8.10.6 Colors, Color Names & Codes

You can use color names in several configuration options and in some internal commands. The general form of a color specification is:

```
[BRIGHT] fg ON [BRIGHT] bg
```

where ***fg*** is the foreground or text color, and ***bg*** is the background color.

### Color Names

Color names as well as the attribute name **BRIGHT** may be shortened to their first three letters. The available color names, shown below on approximations of the 8 basic background colors, are: **BLAck**, **BLUe**, **GREen**, **CYAn**, **RED**, **MAGenta**, **YELLOW**, **WHIte**.

	BLUe	GREen	CYAn	RED	MAGenta	YELLOW	WHIte
BLAck	BLU	GREen	CYAn	RED	MAGenta	YELLOW	WHIte
BLAck	BLUE	GRE	CYAn	RED	MAGenta	YELLOW	WHIte
BLAck	BLUE	GREEN	CYA	RED	MAGenta	YELLOW	WHIte
BLAck	BLUE	GREEN	CYAn	RED	MAGenta	YELLOW	WHIte
BLAck	BLUE	GREEN	CYAn	RED	MAG	YELLOW	WHIte
BLAck	BLUE	GREEN	CYAn	RED	MAGenta	YEL	WHIte
BLAck	BLUE	GREEN	CYAn	RED	MAGenta	Yellow	WHI

**Note:** The colors (if any) represented by your viewer in the above table do not necessarily match the actual rendition provided by your display hardware and drivers at a **TCC** prompt. **BRIGHT** backgrounds are generally always enabled under Windows.

### Color Codes

You can also specify colors by numeric code (see table below) instead of by name. The numeric form is most useful in potentially long options such as [ColorDIR](#), where using color names may take too much space. The codes are decimal numbers, with the codes for bright colors larger than those of the corresponding normal colors by 8.

The [COLOR](#) command also supports the CMD style color specification ***bf***, where ***b*** and ***f*** are CMD.EXE's codes for background and foreground colors, respectively (shown in the CMD columns of the table below). The numeric values of these codes are the same as the **TCC** codes, but they are represented in hexadecimal.

**ANSI X3.64** color codes are also shown in the table. Note that X3.64 support for the *bright* attribute is restricted to foreground. Note that the color codes are decimal, and the codes for *background* colors are

larger than those of the corresponding *foreground* colors by 10.

SCREEN COLOR		TCC name	TCC codes (decimal)		CMD codes* (hexadecimal)		ANSI X3.64 codes (decimal)	
normal	bright		norma l	bright	normal	bright	foreground	background
black	gray	BLAck	0	8	0	8	30	40
blue	blue	BLUe	1	9	1	9	34	44
green	green	GREen	2	10	2	A	32	42
cyan	cyan	CYAn	3	11	3	B	36	46
red	pink	RED	4	12	4	C	31	41
magenta	magenta	MAGenta	5	13	5	D	35	45
brown	yellow	YELLOW	6	14	6	E	33	43
white	white	WHIte	7	15	7	F	37	47

**Note:** The numeric values of the CMD and native color codes are identical, the difference is in representation only.

Use one number to substitute for the [BRIght] fg portion of the color name, and a second to substitute for the [BRIght] bg portion. For example, instead of **bright white on red** you could use **15 on 4** to save space in a ColorDir specification.

The [@OPTION](#) function returns the value of color configuration options by combining both foreground and background into a single number (0-255) using the following logic:

foreground value + ( background value \* 16 ) = code

For example, **bright white on red** (15 on 4) can be expressed as:

15 + ( 4 \* 16 ) = 79

The following batch file translates a combined numeric color code:

```
@echo off
setlocal
function x=%@if[%1 gt 8,bri ,]%@word[%@eval[%1 %% 8],bla blu gre cya red
 mag yel whi]`
:loop
input /c /d ^nColor code? %%c
if %c gt 255 .or. %c lt 0 quit
set f=%@eval[%c %% 16] & set b=%@eval[%c \ 16]
echos The color code %c is "%f on %b" ("%@x[%f] on %@x[%b]")
goto loop
```

## Color Errors

A standard color specification allows sixteen foreground and sixteen background colors. However, many monitors do not provide true renditions of certain colors. For example, most users see normal "yellow" as brown, and bright yellow as yellow; many also see normal red as red, and "bright red" as pink. Color errors are often worse when running in windowed mode, because Windows may not map the text-mode

colors the way you expect. These problems are inherent in the monitor and they cannot be corrected using the **Take Command** color specifications. You can, however, define a custom color palette to get the exact colors you want, via the "Tab Colors" button on the [Configure Take Command / Tab](#) dialog.

## 9 Copyright & Version



**Take Command** 16.03 for Microsoft Windows Vista / Server 2008 / Windows 7 / Windows 8 / Server 2012

Software: Copyright © 2014, Rex Conn and JP Software Inc.  
All Rights Reserved.

Version 16.03 Help System  
Help text: Copyright © 2014 JP Software Inc.  
All Rights Reserved.

Language translations by Christian Albaret (French), Hans-Peter Grözinger (German), Stefano Piccardi (Italian), Dmitry Yerokhin (Russian), and Orlando Hevia (Spanish).

VIEW is a licensed version of the V file viewer, Copyright 1996-2013 by Charles Prineas <http://www.fileviewer.com>

TPIPE is a licensed version of the TextPipe Engine <http://datamystic.com>

The Scintilla edit control is Copyright 1998-2013 by Neil Hodgson <http://www.scintilla.org>

We gratefully acknowledge the contributions of Charles Dye, Vincent Fatica, Steve Fabian, and our other users.

This help material was last revised on Thursday, June 12, 2014

---

*Take Command ® is a registered trademark of JP Software Inc. JP Software, jpsoft.com, and all JP Software designs and logos are also trademarks of JP Software Inc. Other product and company names are trademarks of their respective owners.*

# Index

## - ! -

! 610  
! range exclusion 823  
!~ regular expression inequality test 770  
!= inequality test operator 770

## - \$ -

\$ metacharacter 344  
\$ parameter 801

## - & -

& 221  
& ampersand 789  
&& 782

## - ( -

() parentheses 770, 783

## - \* -

\* (disable alias) 782  
\* (wildcard) 813  
\* parameter 801

## - . -

.AND. 770  
.BAT extension 800  
.BTM extension 800  
.CMD extension 800  
.INI 620  
.INI files 126, 845  
.OR. 770  
.tcl 812  
.XOR. 770

## - / -

/= 165

## - ? -

? (List commands) 165  
? (variable) 611, 629  
? (wildcard) 813

## - @ -

@ at sign 766, 835  
@ABS 647  
@AFSCELL 647  
@AFSMOUNT 647  
@AFSPATH 648  
@AFSSYMLINK 648  
@AFSVOLID 648  
@AFSVOLNAME 648  
@AGEDATE 648  
@ALIAS 648  
@ALTNNAME 649  
@ARRAYINFO 649  
@ASCII 649  
@ASSOC 178, 275, 650  
@ATTRIB 650, 830  
@AVERAGE 651  
@B64DECODE 651  
@B64ENCODE 651  
@BALLOC 652  
@BFREE 652  
@BPEEK 652  
@BPEEKSTR 652  
@BPOKE 653  
@BPOKESTR 653  
@BREAD 654  
@BWRITE 654  
@CAPI 654  
@CAPS 655  
@CDROM 655  
@CEILING 656  
@CHAR 656  
@CLIP 656  
@CLIPW 657  
@COLOR 657

@COMMA 658  
@COMPARE 659  
@CONSOLE 659  
@CONSOLEB 659  
@CONVERT 659  
@COUNT 660  
@CRC32 660  
@CWD 661  
@CWDS 661  
@DATE 661  
@DATECONV 662  
@DAY 662  
@DEBUG 663  
@DEC 663  
@DECIMAL 663  
@DESCRIPT 663  
@DEVICE 664  
@DIGITS 664  
@DIRSTACK 664  
@DISKFREE 665  
@DISKTOTAL 665  
@DISKUSED 666  
@DOMAIN 666  
@DOW 666  
@DOWF 667  
@DOWI 667  
@DOY 668  
@DRIVETYPE 668  
@DRIVETYPEEX 669  
@ENUMSERVERS 669  
@ENUMSHARES 670  
@ERRTEXT 670  
@EVAL 663, 671, 701  
@EXEC 675  
@EXECARRAY 618, 675  
@EXECSTR 618, 675  
@EXETYPE 676  
@EXPAND 676  
@EXT 677  
@FIELD 677  
@FIELDS 678  
@File List 835  
@FILEAGE 679  
@FILEARRAY 679  
@FILECLOSE 679  
@FILEDATE 680  
@FILEHANDLE 680  
@FILENAME 680  
@FILEOPEN 681  
@FILEREAD 681  
@FILEREADB 682  
@FILES 683  
@FILESEEK 683  
@FILESEEKL 684  
@FILESIZE 685  
@FILETIME 686  
@FILEWRITE 686  
@FILEWRITEB 687  
@FILTER 687  
@FINDCLOSE 688  
@FINDFIRST 688  
@FINDNEXT 689  
@FLOOR 689  
@FOLDERS 690  
@FONT 690  
@FORMAT 691  
@FORMATN 691  
@FORMATNC 692  
@FSTYPE 692  
@FTYPE 178, 275, 693  
@FULL 693  
@FUNCTION 693  
@GETDATE 694  
@GETDATETIME 694  
@GETDIR 694  
@GETFILE 695  
@GETFOLDER 697  
@GROUP 666, 698  
@HEXDECODE 699  
@HEXENCODE 699  
@HISTORY 699  
@HTMLDECODE 699  
@HTMLLENCODE 699  
@IDOW 700  
@IDOWF 700  
@IF 700, 770  
@INC 701  
@INDEX 701  
@INIREAD 702  
@INIWRITE 703  
@INODE 703  
@INSERT 704  
@INSTR 704, 740  
@INT 705  
@IPADDRESS 705  
@IPADDRESSN 705

@IPALIASES 705  
 @IPDESC 706  
 @IPDHCP 706  
 @IPGATEWAY 706  
 @IPIPV6 706  
 @IPNAME 706  
 @IPNAMEN 707  
 @IPPHYSICAL 707  
 @IPPORT 707  
 @IPSERVICEALIASES 707  
 @IPSUBNET 707  
 @IPTYPE 707  
 @IPWINS 708  
 @IPWINSSERVER 708  
 @IPZONEID 708  
 @ISALNUM 708  
 @ISALPHA 709  
 @ISASCII 709  
 @ISCNTRL 710  
 @ISDIGIT 710  
 @ISFLOAT 710  
 @ISLOWER 711  
 @ISODOWI 711  
 @ISO\_WEEK 711  
 @ISOWYEAR 711  
 @ISPRIME 711  
 @ISPRINT 712  
 @ISPROC 712  
 @ISPUNCT 712  
 @ISSPACE 713  
 @ISUPPER 713  
 @ISXDIGIT 713  
 @JUNCTION 714  
 @LABEL 714  
 @LCS 714  
 @LEFT 714  
 @LEN 714  
 @LFN 715  
 @LINE 715  
 @LINES 716  
 @LINKS 716  
 @LOWER 716, 750  
 @LTRIM 717  
 @LUA 311, 717, 810  
 @MACADDRESS 717  
 @MAKEAGE 717  
 @MAKEDATE 718  
 @MAKETIME 718  
 @MAX 719  
 @MD5 719  
 @MIN 719  
 @MONTH 720  
 @MX 720  
 @NAME 720  
 @NUMERIC 721  
 @OPTION 721  
 @OWNER 722, 830  
 @PATH 722  
 @PERL 722, 811  
 @PID 722  
 @PIDCOMMAND 722  
 @PING 723  
 @PLUGINVER 723  
 @PPID 723  
 @PRIME 723  
 @PRIORITY 723  
 @PROCESSTIME 724  
 @PYTHON 724, 811  
 @QUOTE 724  
 @RANDOM 724  
 @READSCR 725  
 @READY 725  
 @REGBREAD 725  
 @REGBWRITE 726  
 @REGCOPYKEY 726  
 @REGCREATE 726  
 @REGDELKEY 727  
 @REGEX 727, 915  
 @REGEXINDEX 727, 915  
 @REGEXIST 727  
 @REGEXSUB 727, 915  
 @REGQUERY 728  
 @REGSET 728  
 @REGSETEENV 728  
 @REGTYPE 729  
 @REMOTE 729  
 @REMOVABLE 729  
 @REPEAT 729  
 @REPLACE 730  
 @REREREPLACE 730  
 @REVERSE 730  
 @REXX 731, 811  
 @RIGHT 731  
 @RTRIM 731  
 @RUBY 624, 731, 812  
 @SCRIPT 732

@SEARCH 732  
@SELECT 363, 732  
@SERIAL 733  
@SERIALPORTCLOSE 733  
@SERIALPORTFLUSH 733  
@SERIALPORTOPEN 733  
@SERIALPORTREAD 734  
@SERIALPORTWRITE 734  
@SERVER 734  
@SERVICE 735  
@SFN 736  
@SHA1 736  
@SHA256 736  
@SHA384 737  
@SHA512 737  
@SHFOLDER 737  
@SIMILAR 738  
@SMCLOSE 738  
@SMOPEN 739  
@SMPEEK 739  
@SMPOKE 739  
@SMREAD 739  
@SMWRITE 740  
@SNAPSHOT 740  
@STRIP 740  
@SUBST 741  
@SUBSTR 704, 740  
@SUMMARY 741  
@SYMLINK 742  
@SYSTEMTIME 742  
@TALNUM 742  
@TALPHA 742  
@TARCFILE 743  
@TARCOUNT 743  
@TARDFILE 743  
@TARFILEDATE 743  
@TARFILESIZE 743  
@TASCII 743  
@TCL 744  
@TCNTRL 744  
@TDIGIT 744  
@TIME 744  
@TIMER 413, 745  
@TK 745  
@TLOWER 745  
@TPRINT 746  
@TPUNCT 746  
@TRIM 746  
@TRIMALL 746  
@TRUENAME 443, 747  
@TRUNCATE 747  
@TSPACE 747  
@TUPPER 748  
@TXDIGIT 748  
@UNC 748  
@UNICODE 748  
@UNIQUE 749, 751  
@UNQUOTE 749  
@UNQUOTES 749  
@UPPER 750  
@URLDECODE 750  
@URLENCODE 750  
@UTF8DECODE 750  
@UTFENCODE 750  
@UUDECODE 750  
@UUENCODE 750  
@UUID 750  
@VERINFO 751  
@VERSION 751  
@WATTRIB 752, 830  
@WILD 752  
@WINAPI 342, 654, 753  
@WINCLASS 754  
@WINCLIENTSIZE 754  
@WINEXENAME 754  
@WININFO 615, 754  
@WINMEMORY 754  
@WINMETRICS 755  
@WINPID 756  
@WINPOS 756  
@WINSIZE 757  
@WINSTATE 757  
@WINSYSTEM 757  
@WINTITLE 759  
@WMI 759  
@WORD 760  
@WORDS 760  
@WORKGROUP 761  
@XMLCLOSE 761  
@XMLNODES 761  
@XMLOPEN 761  
@XMLPATH 761  
@YDECODE 762  
@YEAR 762  
@YENCODE 762  
@ZIPCFILE 762

@ZIPCFILESIZE 762  
@ZIPCOMMENT 762, 763  
@ZIPCOUNT 763  
@ZIPDFILE 763  
@ZIPDFFILESIZE 763  
@ZIPFILECRC 763  
@ZIPFILEDATE 763

- [ -

[ ] (wildcard) 813

- \ -

\ backslash 779  
\x 502

- ^ -

^ caret 786, 789

- - -

\_? 611  
\_4VER 612  
\_7unzip\_errors 589, 609  
\_7unzip\_files 589, 609  
\_7zip\_errors 590, 609  
\_7zip\_files 590, 609  
\_ACSTATUS 612  
\_ADMIN 612  
\_AFSWCELL 612  
\_ALT 612  
\_ANSI 612  
\_attrib\_dirs 180, 609  
\_attrib\_errors 180, 609  
\_attrib\_files 180, 609  
\_BATCH 612  
\_BATCLINE 612  
\_BATCHNAME 613  
\_BATCHTYPE 613  
\_BATTERY 613  
\_BATTERYLIFE 613  
\_BATTERYPERCENT 613  
\_BDEBUGGER 613  
\_BG 613  
\_BOOT 613

\_BUILD 613  
\_CAPSLOCK 614  
\_CDROMS 614, 618  
\_CHILDPID 614  
\_CI 614  
\_CMDLINE 614  
\_CMDPROC 614  
\_CMDSPEC 614  
\_CO 614  
\_CODEPAGE 614  
\_COLUMN 615  
\_COLUMNS 615  
\_CONSOLEB 615  
\_CONSOLEPIDS 615  
\_copy\_dirs 201, 609  
\_copy\_errors 201, 609  
\_copy\_files 201, 609  
\_COUNTRY 615  
\_CPU 615  
\_CPUUSAGE 615  
\_CTRL 615  
\_CWD 616  
\_CWDS 616  
\_CWP 616  
\_CWPS 616  
\_DATE 616  
\_DATETIME 616  
\_DAY 616  
\_del\_dirs 212, 609  
\_del\_errors 212, 609  
\_del\_files 212, 609  
\_DETACHPID 616  
\_dir\_dirs 222, 609  
\_dir\_errors 222, 609  
\_dir\_files 222, 609  
\_DISK 616  
\_DNAME 218, 616  
\_do\_dirs 237, 609  
\_do\_errors 237, 609  
\_do\_files 237, 609  
\_do\_loop 237  
\_DOS 617  
\_DOSVER 617  
\_DOW 617  
\_DOWF 617  
\_DOWI 617  
\_DOY 617  
\_DRIVES 617

\_DST 617  
\_DVDS 614, 618  
\_ECHO 618  
\_EDITMODE 618  
\_ELEVATED 618  
\_EXECARRAY 618  
\_EXECSTR 618  
\_EXIT 618  
\_EXPANSION 618  
\_ffind\_errors 258, 609  
\_ffind\_files 258, 609  
\_ffind\_matches 258, 609  
\_FG 618  
\_for\_errors 266, 609  
\_for\_files 266, 609  
\_FTPPEROR 618  
\_HDRIVES 619  
\_head\_errors 285, 609  
\_head\_files 285, 609  
\_HLOGFILE 619  
\_HOST 619  
\_HOUR 619  
\_HWP PROFILE 619  
\_IDE 619  
\_IDLETICKS 619  
\_IDOW 619  
\_IDOWF 619  
\_IFTP 620, 621  
\_IFTPS 620, 621  
\_IMONTH 620  
\_IMONTHF 620  
\_ININAME 620  
\_IP 620  
\_IPADAPTER 620  
\_IPADAPTERS 620  
\_IPARPPROXY 620  
\_IPDNS 620  
\_IPDNSSERVER 620  
\_IPROUTING 621  
\_ISFTP 621  
\_ISODATE 621  
\_ISODOWI 621  
\_ISOWDATE 621  
\_ISOWEEK 621  
\_ISOWYEAR 621  
\_KBHIT 621  
\_LALT 621  
\_LASTDIR 621  
\_LASTDISK 621  
\_LCTRL 621  
\_LOGFILE 622  
\_LSHIFT 622  
\_md\_dirs 312, 609  
\_md\_errors 312, 609  
\_MINUTE 622  
\_mklink\_errors 314, 609  
\_mklink\_links 314, 609  
\_mklnk\_errors 315, 609  
\_mklnk\_links 315, 609  
\_MONITORS 622  
\_MONTH 622  
\_MONTHF 622  
\_move\_dirs 609  
\_move\_errors 609  
\_move\_files 609  
\_NUMLOCK 622  
\_OPENAFS 622  
\_OSBUILD 622  
\_PARENT 622  
\_pdir\_dirs 333, 609  
\_pdir\_errors 333, 609  
\_pdir\_files 333, 609  
\_PID 623  
\_PIPE 623  
\_PPID 623  
\_RALT 623  
\_RCTRL 623  
\_rd\_dirs 349, 609  
\_rd\_errors 349, 609  
\_READY 623  
\_REGISTERED 623  
\_ren\_dirs 353, 609  
\_ren\_errors 353, 609  
\_ren\_files 353, 609  
\_ROW 623  
\_ROWS 623  
\_RSHIFT 623  
\_RUBYTYPE 624, 731  
\_RUBYVALUE 624, 731  
\_SCROLLLOCK 624  
\_SECOND 624  
\_SELECTED 624  
\_SERIALPORTS 624  
\_SERVICE 624  
\_SHELL 624  
\_SHIFT 624

\_SHORTCUT 625  
 \_SHRALIAS 625  
 \_STARTPATH 625  
 \_STARTPID 625  
 \_STDERR 625  
 \_STDIN 625  
 \_STDOUT 625  
 \_STZN 625  
 \_STZO 625  
 \_sync\_dirs 394, 609  
 \_sync\_errors 394, 609  
 \_sync\_files 394, 609  
 \_SYSERR 625  
 \_tail\_errors 398, 609  
 \_tail\_files 398, 609  
 \_tar\_errors 400, 609  
 \_tar\_files 400, 609  
 \_TCCVER 625  
 \_TCEXIT 626  
 \_TCFILTER 408, 626  
 \_TCFOLDER 626  
 \_TCLISTVIEW 626  
 \_TCSTART 626  
 \_TCTAB 626  
 \_TCTABACTIVE 626  
 \_TCTABS 626  
 \_TIME 626  
 \_touch\_dirs 415, 609  
 \_touch\_errors 415, 609  
 \_touch\_files 415, 609  
 \_TRANSIENT 626  
 \_type\_errors 444, 609  
 \_type\_files 444, 609  
 \_TZN 625, 626  
 \_TZO 625, 627  
 \_UNICODE 627  
 \_untar\_errors 451, 609  
 \_untar\_files 451, 609  
 \_unzip\_errors 453, 609  
 \_unzip\_files 453, 609  
 \_UTCDATE 627  
 \_UTCDATETIME 627  
 \_UTC HOUR 627  
 \_UTCISODATE 627  
 \_UTC MINUTE 627  
 \_UTCSECOND 627  
 \_UTCTIME 627  
 \_VERMAJOR 627  
 \_VERSION 628  
 \_VIRTUALBOX 628  
 \_VIRTUALPC 628  
 \_VMWARE 628  
 \_VOLUME 628  
 \_VXPIXELS 628  
 \_VYPIXELS 628  
 \_WINDIR 628  
 \_WINFGWINDOW 628  
 \_WINNAME 628  
 \_WINSYSDIR 628  
 \_WINTICKS 628  
 \_WINTITLE 628  
 \_WINUSER 628  
 \_WINVER 628  
 \_WOW64 629  
 \_WOW64DIR 629  
 \_X64 629  
 \_XMOUSE 629  
 \_XPIXELS 629  
 \_XWINDOW 629  
 \_YEAR 629  
 \_YMOUSE 629  
 \_YPIXELS 629  
 \_YWINDOW 629  
 \_zip\_errors 586, 609  
 \_zip\_files 586, 609  
 \_zipsfx\_errors 588, 609  
 \_zipsfx\_files 588, 609

- | -

|| 782

- + -

+ plus sign 611, 789

- = -

= equal sign 611, 789  
 =~ regular expression equality test 770  
 == equality test operator 770

- 2 -

2UP 562, 569

2UP Font 547  
2UP Printing 569

## - 4 -

4NT 545

## - 7 -

7UNZIP 589, 590  
7ZIP 589, 590

## - A -

ABOVENORMAL 388  
AC line status 612  
ACTIVATE 166  
ActiveTcl 812  
Add To Favorites 551  
AddFile 854  
adding columns 521  
adding columns to Grid 524  
Adding Keys 575  
Administrator 612  
Advanced 131, 874  
Advanced .INI Directives 861  
AFS 823  
Alias 168, 251, 446, 648, 854  
Aliases 168, 251, 385, 446, 782, 796, 797, 806  
AliasExpand 854  
AliasList 862  
Alphabetic 709  
Alphabetic characters 742  
Alphanumeric 708  
Alphanumeric characters 742  
Alt Key 612, 621, 623  
Alt-255 764, 854  
Alt-Down 764  
Alt-End 764  
Alt-F1 165  
Alt-Home 764  
Alt-PgDn 764  
Alt-PgUp 764  
Alt-Up 764  
AND 782  
ANSI 843, 927, 933, 935  
ANSI character set 486

ANSI X3.64 status 612  
App Paths 924  
appending 522  
appending to the clipboard 522  
AppendToDir 779  
Archive 830, 913  
Archive attribute 650  
Argument 801, 802  
ARP Proxy 620  
array variables 378, 451, 595, 770  
Arrays 675, 679  
ASCII 481, 489, 544, 649, 709, 927, 928  
ASCII characters 743  
ASCII Tables 928  
ASSOC 178, 179, 275, 650, 926  
ASSOCIATE 179  
ATTRIB 180, 830  
Attributes 180, 650, 752, 830, 913  
Auto-Arrange 495  
AutoFirewall 862  
Auto-Load Grids 527  
Automatic directory change 795  
Automatically Loading Grids 527  
AutoProxy 862  
auto-scrolling 506  
auto-sum columns 521

## - B -

Background Color 613, 935  
background text 490  
Backspace 764, 850  
backward 498  
backwards search 505  
Batch 613  
Batch Debugger 184, 192, 619, 808, 880  
Batch File Delimiters 863  
Batch file name 613  
Batch File Parameter 801  
Batch File Parameters 383  
Batch Files 184, 612, 796, 799, 800, 801, 804, 805, 806, 808, 810  
Batch Line Number 612  
BATCOMP 183, 810  
Battery 613  
Battery charge 613  
BDEBUGGER 184, 192, 808, 880  
BEEP 190, 339

BeginLine 850  
 BELOWNORMAL 388  
 binary file 474  
 binary files 485  
 binary search 502  
 Binary/Raw Printing 570  
 Bksp 764, 850  
 block marker color 490  
 block marking 516  
 BMP 740  
 book mode 569  
 bookmark lines (search bar) 514  
 bookmarks 184, 511  
 bookmarks (copy to clipboard) 511  
 bookmarks (numbered) 512  
 Boot drive 613  
 BOTTOM 166, 583  
 BREAK 191, 327, 805  
 BREAKPOINT 184, 192  
 browser 512  
 Build 613  
 byte 497

## - C -

calendar 694  
 CALL 192  
 CANCEL 193, 349  
 Caps Lock 300, 614  
 carriage control 539  
 carriage return 477, 488, 544  
 Cascade File Windows 493  
 CASE 392  
 Case sensitive filenames 864  
 Case Sensitivity 790  
 CCTYPE 539  
 CD 194, 196  
 CDD 196, 864  
 CDPATH 595, 791  
 CD-ROM 655  
 Cell Name 647  
 center text 503  
 centimetres 568  
 Change TCMD.INI 866  
 Character Device 664  
 character set 481  
 CHCP 198  
 CHDIR 194

Check for Updates 880  
 Child Process ID 614  
 chunks 474, 484, 533, 542  
 chunks and goto 510  
 clear highlight 516  
 clear marked blocks 516  
 clearing selected text 519  
 ClearKeyMap 862  
 Click here for details on Sorting User Commands 557  
 Clipboard 119, 199, 521, 522, 656, 657, 852  
 clipboard (copying wrapped lines) 540  
 clipboard append 522  
 clipboard contents (viewing) 490  
 CLIPMONITOR 199  
 CLOSE 166  
 Close shared memory 738  
 CLS 199  
 CMD 907  
 CMD Compatibility 800, 863, 907  
 CMD.EXE 545, 598, 800, 902  
 CMD.EXE delayed expansion 907  
 CMD.EXE variables 599  
 CMDBatchDelimiters 863  
 CMDLINE 595  
 CMDVariables 863, 907  
 Code Page 198, 614  
 COLOR 200  
 Color Codes 935  
 Color Dialog 657  
 Color Names 935  
 Color settings 199, 200  
 COLORDIR 596  
 Colors 613, 618  
 colors (Highlight All) 490  
 column and line number 519  
 column copy 520  
 Column Fixing 486  
 column marking 520  
 column number 509  
 Column Position 482  
 Column Search 504  
 column sum 521  
 Columns 615  
 COM Interface 361  
 COM1 733, 734  
 COM1: 624  
 command dialog 353

Command Dialogs 165, 406  
Command editing 764  
Command groups 783  
Command history 288, 699, 766, 768, 769, 854, 856, 857, 871  
Command Input Window 114  
Command Line 614, 763, 764, 789, 871  
Command Line Editing Keys 848, 853  
Command Line Search 463  
Command names 770  
Command parsing 787, 802  
Command processor 545, 614  
Command processor exit codes 155  
Command processor options 149  
Command processor path 614  
Command Processor Version 612, 625  
Command separator 611  
Command Variables 609  
CommandEscape 854  
Commands 155, 804  
Commands By Category 159  
Commands By Name 155  
CommandSep 611, 782, 789  
Comparing TCC and TCC/LE 135  
Comparison 770  
    case insensitive 770  
    case sensitive 770  
    numeric 770  
    string 770  
Compatibility 789  
Compiled batch files 183  
Compound Character 782  
Compressed 830, 913  
Compressed attribute 650  
Compressed batch file 613  
Compression 810  
Computer Name 628  
COMSPEC 129, 596  
Conditional commands 782  
Conditional expression 770  
Configuration 125, 330, 378, 844  
Configuration Dialog 127, 866  
Console Font 266, 690  
Console Window 659  
ConsolePopupWindows 863  
Contact 888  
Context Menus 118  
Continuous scrolling 506  
Control characters 744  
Control Key 621, 623  
COPY 201, 317, 599  
Copy (directive) 850  
copy to clipboard 540  
COPYCMD 599  
Copying Selected Text 119  
copying text 521  
copying text to a file 522  
copying to the clipboard 521  
CopyPrompt 201  
Copyright 577, 937  
Copyright text 863  
count 504  
Country Code 615  
CPU 615  
crash 470  
Create Directory 312  
Create shared memory 739  
Creating Grids 524  
CSV delimiter 533  
CSV Export 533  
CSV files 533  
Ctrl key 615  
Ctrl-A 764, 778, 855  
Ctrl-Bksp 764, 851  
Ctrl-Break 191, 764, 805  
Ctrl-C 191, 764, 805  
Ctrl-D 766, 854, 861  
Ctrl-Down 766, 856  
Ctrl-E 766, 854, 861  
Ctrl-End 764, 850  
Ctrl-Enter 766, 855, 861  
Ctrl-F 764, 854  
Ctrl-F1 288, 764, 855, 889  
Ctrl-Home 764, 850  
Ctrl-K 764, 766, 857  
Ctrl-L 764, 851  
Ctrl-Left 764, 853  
Ctrl-PgUp 854, 855  
Ctrl-R 764, 851  
Ctrl-Right 764, 853  
Ctrl-shift right 764  
Ctrl-shift-ins 764  
Ctrl-shift-left 764  
Ctrl-Shift-Tab 854  
Ctrl-Tab 856  
Ctrl-Up 766, 857

Ctrl-V 852  
Ctrl-X 764, 786, 789  
Ctrl-Y 764, 850  
Current command line 614  
Current Date 567  
current file position 511  
Current Line Marker 506  
Current Mapping 544  
current position 498  
Current Working Directory 616, 661  
Cursor 614  
Cursor Column 615  
Cursor Position 360, 615  
Cursor shape 614  
CursorIns 614  
CursorOver 614  
cut 521

## - D -

Date 209, 412, 621, 661  
date / time picker 694  
Date Format 540  
Date Formats 209, 646, 661, 790  
date picker 694  
Date ranges 823, 826  
DATEMONITOR 210  
DATETIME 237  
Day  
    of month 616  
    of week 617  
    of week (full) 617  
    of week (integer) 617  
    of week (localized) 619  
    of year 617  
Day of Month 662  
Day of Week 666, 667, 700  
Day of Year 668  
Daylight Savings Time 617  
Debug 864  
Debug Format 489  
Debug Mode - User Commands 555  
Debugging 184, 808  
Debugging User Commands 558  
DEBUGMONITOR 211  
DEBUGSTRING 212  
decimal 497  
DEFAULT 392

Default Grid Directory 527  
default header format 567  
Default Mapping 544  
default printer 562  
default printer profile 573  
Default User Command 558  
Default Variables 251, 373, 449  
DEFER 212  
DEFINED 770  
DEL 212  
Del (directive) 850  
DELAY 217  
Delayed Variable Expansion 834  
Delete 764  
DELETE wipe 864  
DelHistory 854  
DELIMS (FOR command) 266  
DelToBeginning 850  
DelToEnd 850  
DelWipePasses 864  
DelWordLeft 851  
DelWordRight 851  
DESCRIBE 218, 616  
Description ranges 823  
DescriptionName 218, 616  
Descriptions 830, 874  
Desktop 221, 384  
Desktop Window 740  
DETACH 221, 616  
Detecting 806  
Dialog 121, 122  
DIR 222, 333, 599  
DIRCMD 599  
Directories 911  
Directory 312, 779, 830, 913  
Directory Aliases 796  
Directory attribute 650  
Directory Dialog 694  
Directory history 234, 769, 780, 854, 871  
Directory Navigation 235, 341, 346, 790, 792, 795  
Directory Searches 194, 196, 598, 791, 792  
Directory Stack 235, 341, 346, 664, 790  
DIREXIST 770  
DIRHISTORY 234  
DIRS 235, 341, 346  
DirWinOpen 854  
Disable 378  
Disk serial number 733

DISKMONITOR 236  
Display a Hex 544  
Display Line Number 1 548  
Display Line Numbers 548  
Display Resolution 357  
DNS 620  
DNS Server 620  
DO 237, 770  
DO (FOR command) 266  
DO UNTIL 770  
DO WHILE 770  
DOS Character Set 486  
double DWord 497  
Double quotes 724, 749  
double sided printing 562  
double-click text selection 516  
double-sided printing 466  
Down 766, 851, 856  
Drag and drop 121  
DRAWBOX 242  
DRAWHLINE 243, 244  
DRAWVLINE 243, 244  
Drive 909  
Drive Type 668, 669  
duplex 466  
duplexing 562  
Duplicate CMD.EXE bugs 907  
DWord 497

**- E -**

EBCDIC 481, 544  
EBCDIC Files 535  
EBCDIC fixed record length 537  
ECHO 245, 247, 618, 800  
ECHOERR 246, 248  
Echoing 800  
ECHOS 245, 247  
ECHOERR 246, 248  
ECHOX 248, 249  
ECHOXERR 248, 249  
Edit Menu 106  
Editing 764, 871  
Editing keys 764  
editor 545  
Editor Options 545  
Editor Path 545  
EJECTMEDIA 249

ELSE 290, 291  
ELSEIFF 291  
Enable 378  
Encrypted 830, 913  
Encrypted attribute 650  
Encrypted batch file 613  
End 764, 851  
End of Line 472, 477  
End of Line character 477  
End Of Line Indicator 547  
end offset 519  
end point 516  
ENDDO 237  
EndHistory 854  
ENDIF 291  
EndLine 851  
ENDLOCAL 249, 382  
end-of-line 544  
ENDSWITCH 392  
ENDTEXT 411  
Enter 764, 851, 861  
Environment 251, 373, 449, 593  
Environment Variables 804  
EOL 472, 477, 547  
EOL (FOR command) 266  
EQ 770  
EQC 770  
EQL 770  
EQU 770  
ERASE 212  
EraseLine 851  
Error 327, 625  
error message 503  
Error Messages 890  
error reports 470  
Error Text 670  
ERRORLEVEL 327, 378, 611, 629, 770  
ERRORMSG 327  
Errors 890, 927  
Esc 851  
Escape 611, 854  
Escape character 786  
ESCAPE key 540, 548  
EscapeChar 611, 786, 789, 801  
ESET 251, 446, 596, 597  
even page printing 466  
EVENTLOG 252  
EVENTMONITOR 253

EVERYTHING 255  
Everything Search 255, 864  
EverythingSearch 864  
EXCEPT 256  
exception 470  
Exclusion ranges 823  
ExecLine 851  
Executable extensions 817  
Executable Files 924  
ExecWait 786  
EXIST 770  
EXIT 257, 618  
Exit Code 155, 611, 782  
Expand Aliases 854  
Export Data 530  
Export Favorites 554  
Export keys 576  
Export to CSV 533  
Export User Commands 561  
Exporting Grids 530  
Extended directory search 871  
Extended Directory Searches 194, 196, 346, 598  
Extended Parent Directory Names 779, 834  
EXTPROC 812

## - F -

F1 288, 764, 855, 889  
F10 854  
F12 857  
F3 766  
F6 854  
F7 856  
F8 856, 864  
F9 856  
FAT 910  
FAT32 910  
Favorite Files 553  
Favorite Searches 514  
Favorites 550  
Favorites (Export) 554  
Favorites Organize 552  
FF 568  
FFIND 258  
File Attributes 488  
file chunks 533, 542  
File Chunks (overview) 474  
file colors 490  
File completion 856, 857  
File Dialog 695  
File Encoding 522  
File exclusion ranges 829  
File Extension 677  
File Filters 418  
File List 472, 835  
file locking (overview) 471  
File Names 909, 912  
file offset 509  
File Position Maintenance 513  
File Prompts 844  
File Searches 834, 924  
File selection 813, 836  
File Streams 915  
File Systems 909, 910  
File Tailing 489  
File Tailing (overview) 474  
File Time Stamps 914  
File View Split 491  
File Windows - Multiple 493  
FILECOMPLETION 596  
FileCompletion directive 777, 778  
FILECOMPLETION variable 777, 778  
FileCompletionLooping 864  
Filename completion 775, 777, 778, 871  
Filename conversion 778  
Filenames 778  
FilesCaseSensitive 864  
FILL 242  
filter 466  
find 498  
Find Next 503  
Find Next from Current 503  
Find Next from next line 503  
Firewall 877  
FIREWIREMONITOR 263  
Fix Columns 486  
Fix Line Numbers 486  
fixed length records 476, 509, 522, 527  
fixed line length 479  
Flat Text mode 476  
flip ends 497  
Floating Ruler 482  
Folder Dialog 697  
Folder Locations 737  
Folder View from TCC 134  
FOLDERMONITOR 264

Folders 113  
 FONT 129, 266, 498, 690  
 Font Zoom 472  
 Fonts 486, 547  
 footer 567  
 FOR 266  
 Foreground Color 618, 935  
 Foreground Window 628  
 FOREVER 237  
 form feed 568  
 Form Feeds 513, 562, 568  
 Format Number 691  
 format specifiers 567  
 Format Text 691  
 found color 490  
 found text 490  
 FREE 275  
 Frequency 190, 339  
 FS 388  
 FTP 293, 618, 818  
 FTP options 877  
 FTP.CFG 818  
 FTPS 293, 618, 818  
 FTYPE 178, 179, 275, 693, 926  
 Full Path Name 567  
 FUNCTION 276, 447  
 Functions 593, 630, 632  
 Functions by Category 639  
 Functions Dialog 276

**- G -**

GE 770  
 General Input Keys 848, 849  
 Generic/Text Only 571  
 GEQ 770  
 Global 280, 769  
 global alias size 862  
 Global aliases 168  
 GOSUB 281, 357  
 Goto 283, 291, 509  
 goto and chunks 510  
 Greenbar Colors 487  
 Greenbar Options 487  
 Grid Export 530  
 Grid Import 532  
 grid line color 490  
 Grid Lines 482

Grid rename 526  
 Grid Rules 529  
 GridLines 522  
 GridLines - automatically loading 527  
 GridLines - Default Directory 527  
 GridLines (printing) 567  
 GridLines creating 524  
 GridLines Fixed length records 527  
 GridLines Organize 526  
 Gridlines Wrap Options 527  
 Grids (export) 554  
 Grids and file extensions 530  
 GT 770  
 GTR 770  
 GUID 750  
 GUNZIP 284  
 GZIP 284, 448

**- H -**

Hard Link 314, 315  
 Hardware Profile 619  
 HEAD 285, 398, 444  
 header 567  
 Headers and Footers 562, 567  
 HELP 288, 855, 889  
 Help Menu 111  
 HelpWord 288, 855, 889  
 here-document 838  
 hex 489  
 hex characters 502, 503  
 Hex Editor 545  
 hex font 498  
 hex format 497  
 hex line length 498  
 hex mode 489, 495  
 hex offset 519  
 hex offset in status bar 540  
 hex search 498, 502  
 hex string 502  
 Hexadecimal 748  
 Hidden 830, 913  
 Hidden attribute 650  
 HIDE 166, 583  
 HIGH 388  
 highlight 516  
 Highlight All (Search Bar) 514  
 Highlight All colors 490

- Highlighting Text 119  
 HistLogName 310  
 HistLogOn 310  
 History 288, 310, 542, 596, 699, 854, 855, 856, 857  
 History buffer sizes 871  
 History Log File 619  
 HISTORYEXCLUDE 596  
 HistWinOpen 855  
 Home 764, 850  
 HomeMenu 105  
 Host name 619  
 hotspot 512  
 hour 619  
 HP LaserJet 547, 569  
 HTML decoding 699  
 HTML encoding 699  
 HTTP 512, 818  
 HTTP firewall 862  
 HTTP proxy 877  
 HTTP proxy server 862  
 http://jpsoft.com/ 111  
 HTTPS 818  
 hyperlink 512
- I -**
- IBM 481  
 IDE 184, 619, 880  
 IF 290, 700, 770  
 IFF 290, 291, 770  
 IFTP 293, 618, 620, 621, 818  
 IM 299  
 Import Favorites 554  
 Import Keys 576  
 Importing Grids 532  
 inches 568  
 Include lists 833  
 Indirect file 835  
 INIQuery 865  
 Initialization 126, 845  
 Initialization files 126, 845, 846  
 INKEY 295, 297  
 Inode 703  
 In-Process Pipe 841  
 INPUT 295, 297, 347  
 Input redirection 843  
 Ins 852  
 Insert 614, 764, 852  
 Insert cursor 614  
 Insert cursor shape 614  
 Instant Message 299  
 Integration 134  
 Integrity attribute 830  
 IntelliMouse 508  
 Internal Commands 155, 159  
 Internal Variables 600, 601, 605  
 Internet 293, 818, 877  
 Introduction 98  
 INV 388  
 IP 705, 706, 707, 708  
 IP Adapter 620  
 IP Adapters 620  
 IP Address 620  
 ipworks6.dll 818  
 ipwssl6.dll 818  
 ISALIAS 770  
 ISAPP 770  
 ISDIR 770  
 ISFILE 770  
 ISFUNCTION 770  
 ISINTERNAL 770  
 ISLABEL 770  
 ISO 8601 790  
 ISO date 616  
 ISWINDOW 770  
 ITERATE 237
- J -**
- JABBER 299  
 JABBER options 877  
 JavaScript 361  
 JP Software 888  
 jphelp.chm 889  
 JPSTREE.IDX 194, 196, 346, 598, 792  
 Junction (reparse point) 314, 315, 714, 830, 913  
 Junction (reparse point) attribute 650
- K -**
- Key Codes 927  
 Key Mapping Directives 764, 766, 768, 775, 848  
 Key Names 932  
 KEYBD 300  
 Keyboard 300, 621

Keyboard Customization 573  
Keyboard Shortcuts 117, 573  
Keypad 764, 927  
KEYS 301, 932  
Keys (Export/Import) 576  
keys for searching 505  
KEYSTACK 301, 843

## - L -

Label 577, 714, 913  
Landscape 569  
Language 127, 865  
LanguageDLL 865  
large files 474  
LastHistory 855  
LE 770  
LEAVE 237  
LEAVEFOR (FOR command) 266  
Left 764, 852  
Length limits 789  
LEQ 770  
LFN 778, 834, 855, 912  
LFNToggle 778, 855  
Limits 908  
line and column number 519  
Line Continuation 801  
Line Drawing characters 486  
line feed 477, 488, 544  
line length 498  
Line Lengths 488  
line number 509  
Line Number Increment 548  
Line Number Reset 548  
Line Numbers 484, 548  
Line Numbers in Chunks 484  
Line Range (print) 562  
line terminator 477, 488  
line wrap 565  
Line Wrapping 485  
LinePrinter 547  
LinePrinter font 569  
LineToEnd 855  
Link 314, 315  
LIST 303, 457, 858, 859, 860  
LIST Keys 848, 858  
List of Keys 576  
List View 113

List View selection 626  
ListBack 858  
ListClipboard 858  
ListContinue 858  
ListExit 858  
ListFind 859  
ListFindRegex 859  
ListHex 859  
ListHighBit 859  
ListInfo 859  
ListNext 859  
ListOpen 859  
ListPrevious 860  
ListPrint 860  
ListRefresh 860  
ListUnicode 860  
ListWrap 860  
LOADBTM 309  
LOADMEDIA 310  
Local 769  
Local lists 867  
LocalAliases 168  
Localization 874  
LOG 310  
Log File 622  
Log Off 351  
LogErrors 310  
Logging options 867  
Logical expression 770  
Logical operator 770  
LogName 310  
LogOn 310  
Long File Name 834  
Long Filename 778  
long lines 485  
Longest Common Sequence 714  
longest line 488  
LOW 388  
Lower Case 716  
Lower case characters 745  
LSS 770  
LT 770  
Lua 311, 810  
Lua Directive 865  
Lua Scripts 865

**- M -**

MAC address 717  
MailAddress 369  
MailPassword 369  
MailPort 369  
MailServer 369  
MailUser 369  
mapping 481  
Margins 562, 568  
mark blocks 516  
marker color 490  
marking columns 520  
match case toggle 505  
MAX 166, 388, 583  
MD 312  
MEMORY 314  
Menus 103, 105, 106, 108, 110, 111  
Message Box 323  
Metacharacters 344  
middle mouse button 540  
MIN 166, 388, 583  
Minimize to Tray 127  
Minute 622  
MKDIR 312  
MKLINK 314  
MKLNK 315  
MONITOR 388  
Monitor Resolution 357  
Month 620, 622, 720  
more 466  
More Options 542  
More? 783  
Mount Point 647  
Mouse column position 629  
Mouse position 629  
MOVE 201, 317  
Moving Favorites 552  
MRU 542  
MSAA 865  
MSAAMenu 865  
MSDOS File Name 545  
MSGBOX 323  
Multiple Commands 782  
Multiple File Windows 493  
Multiple filenames 832

**- N -**

Navigation 598  
NE 770  
NEQ 770  
Nesting Level 612  
NETMONITOR 326  
Network Drive 729  
Network Routing 621  
new line 477  
New Page 513, 568  
next 472  
NextFile 856  
NextHistory 856  
No scrub data attribute 830  
NoNIErrors 865  
non-proportional font 547  
Normal 388, 830, 913  
Normal attribute 650  
NormalEditKey 856  
NormalKey 852  
NormalListKey 860  
NormalPopupKey 861  
NOT 770  
Not content-indexed 830, 913  
Not content-indexed attribute 650  
not found 503  
NOTOPMOST 166, 583  
NTFS 910, 915  
NTFS Links 716  
NTFSDescriptions 218  
Num Lock 300  
numbered bookmarks 512  
Numeric 744  
NumLock 622  
  
**- O -**  
octal 497  
odd page printing 466  
OEM Character Set 486  
Offline 830, 913  
Offline attribute 650  
offset 489, 509  
ON 327, 805  
Online Help 889

Open Selection in Browser 512  
 Open Selection in V 512  
 OpenAFS 612, 622, 647, 648, 823, 910  
 OPTION 330, 721  
 Options 127, 129, 131, 461, 867, 870, 871, 874, 877  
 Options Menu 110  
 OR 782  
 Organize Grids 526  
 Organizing Favorites 552  
 Organizing User Commands 557  
 OSD 331  
 OutputDebugString 211  
 Overstrike 614, 852  
 Overstrike cursor shape 614  
 Overview 1  
 over-writing 522  
 Owner ranges 830

**- P -**

Pad with zero 548  
 Page and file prompts 844  
 page break 568  
 page breaks in EBCDIC files 539  
 Page Length 562, 568  
 page limit 466  
 page marker 513  
 Page Number 567  
 Page prompts 844  
 Page Range (print) 562  
 Page Up/Down 548  
 pages 513  
 pagination 513  
 Parameter 802  
 Parameter quoting 802  
 ParameterChar 789, 801  
 Parameters 770, 801  
 Parent Directory 779, 834  
 Parent process 622  
 Parsing 787, 802  
 Paste 119, 852  
 Path 332, 596, 722  
 PATHEXT 596  
 PAUSE 333  
 pause scrolling 506  
 PDIR 222, 333  
 Perl 722, 811

PerlScript 361  
 PGM 388  
 PgUp 855  
 Ping 723  
 pipe 466  
 pipe to VIEW 461  
 Pipes 837, 841  
 Piping 837  
 Pixels 629  
 Platforms 889  
 PLAYAVI 338  
 PLAYSOUND 190, 339  
 PLUGIN 340  
 POPD 235, 341, 346  
 PopFile 856  
 Pop-up Font 127  
 Pop-up window font 870  
 Popup Window Keys 848, 860  
 Popup Windows 926  
 PopupWinDel 861  
 PopupWinEdit 861  
 PopupWinEditWin 861  
 PopupWinExec 861  
 POS 166, 388, 583  
 POST\_EXEC 168  
 POSTMSG 342  
 PRE\_EXEC 168  
 PRE\_INPUT 168  
 precision 671  
 Preferences 539  
 PrevFile 856  
 PrevHistory 857  
 previous 472  
 Primary 624, 846  
 Print 342, 562  
 Print First Line in Page 548  
 Print From Here 566  
 Print from Top of Page 566  
 Print Last Line in Page 548  
 print limit 466  
 Print Line Numbers 562  
 Print New Page 568  
 Print Options 562  
 Print Range 566  
 Print ruler 567  
 Print Setup 562  
 printable character 489  
 Printable characters 746

printer font 547, 568  
Printer Fonts 569  
Printer Profiles 573  
printing gridlines 567  
Printing Line Numbers 548  
printing the ruler 567  
PRIORITY 343  
Process ID (PID) 221, 404, 405, 614, 616, 623, 625  
PROCESSMONITOR 343  
Profiles (printer) 573  
PROMPT 344, 597  
Prompt before reloading 540  
PROMPT2 597  
Prompting in TCMD.INI 865  
proportional fonts 547, 569  
Punctuation characters 746  
PUSHD 235, 341, 346  
Python 724, 811

## - Q -

QUERYBOX 347  
QUIT 193, 349  
Quotes 724, 749  
Quoting 802

## - R -

RAM 314  
Random 724  
Ranges 823, 825, 826, 828, 829, 830  
Raw/Binray Printing 570  
RD 349  
Read shared memory 739  
Read-only 830, 913  
Read-only attribute 650  
REALTIME 388  
REBOOT 351  
Recall 766  
Recent Files 542  
RECFM 536  
RECFM ASCII files 536  
RECFM=F 537  
RECFM=U 538  
RECFM=V/VB 536  
record number 509  
RECYCLE 352

Recycle Bin 212, 349, 352, 597, 867  
RECYCLEEXCLUDE 597  
redirected output 461, 466  
Redirection 625, 837, 838  
Redirection and Piping 837  
Redo 857  
Reference 897, 924, 927  
Refresh File 472  
Register 133, 896  
Registration 133, 896  
Registry 251, 373, 449  
    Copy 726  
    Create 726  
    Delete 727  
    Exists 727  
    Query 728  
    Set 728  
    Set (broadcast) 728  
Regular Expression search option 498  
Regular expression syntax 874  
Regular Expressions 727, 813, 915  
Regular Expressions @REREPLACE 730  
Relational expression 770  
Relational operator 770  
REM 352  
Remark 352  
remember file position 513  
Remote Drive 729  
Removable Drive 729  
Removable Media 249  
Remove Directory 349  
REN 317, 353  
RENAME 353  
rename Grid 526  
RepeatFile 857  
Reset Line Numbers 548  
Resizing 120  
RESOLUTION 357  
RESTORE 166, 583  
Restore wrap settings 527  
RETURN 281, 357  
RExec 358, 359  
RExec options 877  
REXX 731, 811  
Right 764, 852  
Right To Left 475  
RMDIR 349  
Row 623

Rows 623  
 RSHELL 358, 359  
 RSHELL options 877  
 RTL 475  
 Ruby 624, 731, 812  
 Ruler 482, 540, 567  
 ruler (printing) 567  
 Run Program Dialog 122  
 RUNAS 388

**- S -**

Save Window 740  
 SaveHistory 857  
 saving text to a file 522  
 Scan Codes 927  
 SCREEN 360, 362  
 Screen Font 547  
 Screen Readers 865  
 Screen resizing 120  
 Screen saver 361  
 Screen Size 629  
 SCREENMONITOR 361  
 SCRIPT 361  
 ScrLk 624  
 scroll bars 506  
 Scroll Lock 300, 624  
 Scrollback Buffer 119  
 Scrolling 506  
 scrolling (synchronized) in split mode 493  
 scrolling (synchronized) with multiple windows 495  
 scrolling speed 506  
 SCRPUT 360, 362, 578  
 search 498  
 Search (Favorites Searches) 514  
 Search Again 498  
 search backwards 505  
 Search Bar 514  
 Search Bar Options 514  
 search colors 490  
 search count 504  
 search history 498  
 search keys 505  
 search line colors 490  
 search message 503  
 Search Options 503, 550  
 Search Skip 504  
 search string 502  
 search text 502  
 searching 498  
 Searching Columns 504  
 Searching on the Command Line 463  
 Second 624  
 Secondary 624, 846  
 SELECT 363, 732  
 selected text 516  
 selected text color 490  
 selecting text 516, 519  
 selecting text (double click) 516  
 Selecting words 520  
 selection length 519  
 self-extracting executable 588  
 Send Error Report 470  
 SENDHTML 367  
 SENDMAIL 369  
 SEPARATE 388  
 Serial Port 733, 734  
 Serial Ports 624  
 Server Completion 127  
 Servers 669  
 SERVICEMONITOR 372  
 SERVICES 373  
 SET 251, 373, 449, 596, 597  
 SETARRAY 378  
 SETDOS 378, 614  
 SETERROR 378  
 SETLOCAL 249, 382  
 Setting colors 199, 200  
 Setup 886  
 SFN 649, 778, 834, 855, 912  
 SHADOW 242  
 Shape 614  
 SHARED 388  
 Shared memory 738, 739, 740  
 Sharenames 670  
 SHEBANG 812  
 SHIFT 383  
 Shift Key 622, 623, 624  
 Shift right 764  
 Shift-left 764  
 Shift-Tab 856  
 Short file name 736  
 Short Filename 778  
 SHORTCUT 384  
 Shortcuts 117, 384  
 shortest line 488

shortest non-empty 488  
 show tabs 479  
 SHRALIAS 385, 625  
 Shutdown 351  
 Single Instance 127  
 SIZE 388, 583  
 Size ranges 823, 825  
 SKIP (FOR command) 266  
 Skip Search 504  
 smooth scrolling 508, 542  
 SMPP 386  
 SMS 386  
 SMTP 877  
 SNMP 387  
 SNPP 387  
 Soft Link 314, 315  
 Sorting Favorites 554  
 Sorting User Commands 561  
 Sound 190, 339  
 Sparse file 830, 913  
 Sparse file attribute 650  
 Special Character Compatibility 786, 789  
 Special characters 874  
 speed 506  
 Split File View 491  
 Splitter Windows 116  
 Standard Error 625, 837, 838, 841  
 Standard Input 625, 837, 838, 841  
 Standard Output 625, 675, 837, 838, 841  
 START 388, 625, 786  
 start offset 479, 519  
 Start Options 149  
 start point 516  
 Starting applications 785  
 StartTabWait 866  
 Startup 98, 149, 867  
 Startup command 149  
 Startup Directory 625  
 Startup drive 613  
 Startup Options (Take Command) 99  
 Startup options (TCC) 149  
 Startup Tabs 129  
 Status Bar 117, 392, 480, 519  
 Status test 770  
 STATUSBAR 392  
 stderr 246, 248, 249, 838  
 stdin 466, 540, 838  
 stdout 245, 247, 248, 838  
 Stopwatch 413, 745  
 Streams 915  
 string count 504  
 string count (search bar) 514  
 String Processing 808  
 String substitution 599  
 Subdirectories 911  
 Subroutine 281, 357  
 Substrings 599  
 sum 521  
 Supported Platforms 889  
 SwapScrollKeys 766  
 SWITCH 392  
 Switch Desktops 221  
 Switches 836  
 Symbolic link 648, 830  
 symbolic link (reparse point) 742  
 Symbolic links 314, 867  
 SYNC 394  
 Synchronize directories 394  
 synchronized scrolling in split mode 493  
 synchronized scrolling with multiple windows 495  
 Syntax Coloring 184  
 System 830, 913  
 System attribute 650  
 System Errors 927  
 System Metrics 755  
 System Variables 251, 373, 449, 595

**- T -**

Tab 856  
 Tab Completion 864  
 Tab Configuration 129  
 Tab Icons 129  
 Tab Size 129, 540  
 Tab startup wait 866  
 Tab Toolbar Dialog 124  
 Tab Window 626  
 Tab Windows 116  
 Tabbed Toolbar 112  
 Tabs 479  
 Tabs menu 108  
 tail 285, 398, 444, 540  
 tailing 489  
 tailing on Unix drives 489  
 Take Command 13.0 40  
 Take Command 14.0 25

Take Command 15.0 13  
Take Command 16.0 2  
Take Command and TCC Integration 134  
Take Command Dialogs 121  
Take Command Interface 100  
Take Command Menus 103  
Take Command Tabs 129  
Take Command Window 101  
Take Command Windows Configuration 127  
TAR 400, 451  
Tar archives 743  
TASKBAR 402  
TASKDIALOG 403  
TASKEND 404  
TASKLIST 405  
TC Scrollback Buffer Keys 848  
TCC /Q 863  
TCC /U8 866  
TCC/LE 135  
TCC/LE 13.0 40  
TCDIALOG 406  
TCEXIT 153  
TCFILTER 408  
Tcl 744, 745, 812  
TCMD 597  
TCMD.GPF 886  
TCMD.INI 125, 126, 620, 844, 845  
TCMD.INI errors 865  
TCMDVER 597  
TCSTART 153  
TCTOOLBAR 408  
Technical Support 886  
TEE 410, 585  
TEMP 597, 598  
Temporary 913  
Temporary attribute 650  
Temporary file 830  
TEXT 411  
text file (saving to) 522  
Text Mode 474  
Text Only Printing 570, 571  
text selection 519  
TFTP 818  
The File View 471  
THEN 291  
Tile File Windows 493  
Time 209, 412, 626, 744  
Time ranges 823, 828  
Time server 877  
Time Stamps 914  
Time Zone 625, 626, 627  
TIMER 413, 745  
TITLE 166, 415, 583, 598, 628  
TITLEPROMPT 598  
Tk 745, 812  
TMP 597, 598  
TOKENS (FOR command) 266  
Tone 190, 339  
Tool Bar 112, 408  
Toolbar - File View 472  
Toolbar Dialog 124  
TOP 166, 583  
TOPMOST 166, 583  
TOUCH 415  
TPIPE 418  
trailing spaces 536  
TRANS 583  
TRANSIENT 441  
Transient Shell 626  
Transparency 127  
traps 387  
TRAY 583  
TrayHotKey 866  
TREE 441  
Tree view 113  
TREEEXCLUDE 598, 792  
Troubleshooting 886  
TRUENAME 443, 747  
TuFix 522  
two-up 569  
TYPE 398, 444

## - U -

U3 file paths 555  
UCS-2 475  
UNALIAS 168, 251, 446  
UNC 748, 910  
Undo 857  
UNFUNCTION 276, 447  
UNGZIP 448  
Unicode 627, 748  
Unicode files 475  
Unicode search option 498  
Unique File Name 749  
Unix drives (tailing) 489

Unix paths 867  
**UNKNOWN\_CMD** 168, 446, 924  
**UNSET** 251, 373, 449, 596, 597  
**UNSETARRAY** 451  
**UNTAR** 400, 451  
**UNTIL** 237  
**UNZIP** 453, 586  
**Up** 766, 853, 857  
**UPDATE.EXE** 897  
**UpdateINI** 866  
**Updater** 897  
**Updates** 880  
**Upper Case** 748, 750  
**URL** 512, 818  
**URL decoding** 750  
**URL encoding** 750  
**USBMONITOR** 455  
**User** 628  
**User Commands** 555, 557  
**User Commands (Export)** 561  
**User defined functions list size** 864  
**User Variables** 251, 373, 449  
**UTF-16** 475  
**UTF8** 750  
**UTF-8** 866  
**UTF8Output** 866  
**Utilities Menu** 110  
**UUID** 750

**- V -**

**Variable** 632  
**Variable Expansion** 834, 857  
**Variable Functions** 630, 632  
**Variable Functions by Category** 639  
**Variable name completion** 781  
**VARIABLEEXCLUDE** 598  
**VariableExpand** 857  
**Variables** 251, 373, 449, 593, 595, 600, 601, 605, 804  
**VBEEP** 456  
**VBScript** 361  
**VER** 456  
**VERIFY** 457  
**Version** 456, 612, 617, 622, 625, 751, 937  
**Version 10 Features** 84  
**Version 11 Features** 73  
**Version 12 Features** 58

**Version 14** 25  
**Version 15** 13  
**Version 16** 2  
**vertical hex mode** 498  
**very large files** 474  
**VFAT** 910  
**VIEW** 457  
**VIEW Command Line Options** 461  
**Viewing Multiple Files** 493  
**viewing the clipboard contents** 490  
**Virtual Screen** 628  
**VirtualBox** 628  
**VirtualPC** 628  
**VMWare** 628  
**VOL** 577  
**Volatile Variables** 251, 373, 449  
**Volume** 577, 628, 909  
**Volume ID** 648  
**Volume Name** 648  
**VSCRPUT** 362, 578

**- W -**

**WAIT** 388  
**Waiting for applications** 786  
**WAKEONLAN** 579  
**WEBFORM** 579  
**WEBUPLOAD** 580  
**What's New Version 13** 40  
**What's New Version 14** 25  
**What's New Version 15** 13  
**What's New Version 16** 2  
**WHICH** 582  
**WHILE** 237  
**White space** 747  
**whole word only** 498  
**Wildcards** 752, 813  
**WIN** 388  
**Win64** 131  
**Window** 166, 583  
  **Class** 754  
  **Position** 756  
  **Process ID** 759  
  **State** 757  
**Window colors** 870  
**Window Layout** 543  
**Window Title** 628  
**Windows** 870

Windows - Multiple 493  
Windows API 753  
Windows Directory 628  
Windows File Associations 926  
Windows Management Instrumentation 759  
Windows Memory 754  
Windows Parameters 757  
Windows System Directory 628  
Windows System Errors 927  
Windows System Metrics 755  
Windows system tray 866  
Windows Version 628  
WMIQUERY 584  
word 497  
word boundary 485  
Word Sets 542  
WordLeft 853  
WordRight 853  
wordsets 520  
Workgroup 761  
Wrap 472, 485  
Wrap Here 485  
Wrap Length 485  
Wrap Lines at Column 527  
Wrap Lines to Screen 540  
Wrap on Word Boundary 540  
wrap settings restore 527  
Wrap Text 485  
Wrap to Length 485  
Wrap to Screen 485  
Wrap to Start 503  
wrapped lines (copying to clipboard) 540  
Wrapping in flat text mode 485  
Wrapping Lines 565  
Write shared memory 739, 740  
Write to File 522

## - Z -

ZIP 453, 586  
Zip archives 762, 763  
ZIPSFX 588  
ZOOM 242  
Zoom Font 472

## - X -

X3.64 843, 927, 933  
x64 629, 897  
XML 922

## - Y -

Y 410, 585  
Year 629, 762