CT-Shell for Windows

Multiple Configurations

CT-Shell normally starts with a configuration file called CTSHELL.INI, which it expects to find in the current directory, the Windows home directory, in a directory along the DOS executable path, or in a directory that is part of a special network setup. However the program may be started with an optional initialization filespec on the command line, to have it use a different configuration file. Thus, multiple configurations may be used for different purposes.

If you specify an initialization filespec on the command line when you start CT-Shell, be sure to provide a complete path. For example, this command line might start CT-Shell with a configuration that is customized for program development:

CTSHELL c:\win\program.ini

whereas this one would start CT-Shell with a configuration that was customized for word processing projects:

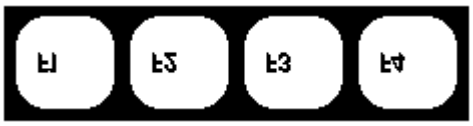
CTSHELL c:\win\wordproc.ini

The CT-SHELL Window

The CT-Shell Window contains a number of features that provide services to the user. They include the Menu, Function Keys, Status Display, Current Path, Files Window and Command Line.

The CT-Shell Menu

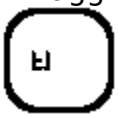
Most routine operations can become entries in your menu, so that a keypress or a mouse click is all that's needed to accomplish them. The CT-Shell menu contains items that each invoke a popup menu, from which individual menu entries are selected using keyboard or mouse methods. You can add entries to your menu by adding entries to the [ITEMS] section in your CTSHELL.INI file.



Function Keys

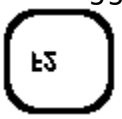
There are pre-assigned meanings for most of the function keys, but <F9> and <F10> have been kept available for the user to define.

Much of what CT-Shell does with files can be done either with a current file or with a set of tagged files. When a file is tagged, its entry in the files list is highlighted. The first five function keys are devoted to managing those file tagging operations.



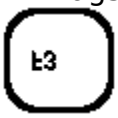
Toggle Current

Toggles the tagged/untagged condition of the current file.



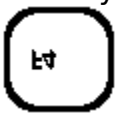
Tag All

Tags all the files (but not directories or drives) that are in the files listing.



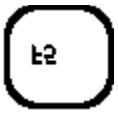
Untag All

Untags all the files, regardless of how many were tagged, or how they got that way.



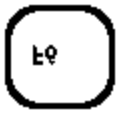
Invert Tags

Inverts all the tags, will tag all the previously untagged files, and untag the ones that were tagged.



Tag By Name

Tags by name. Both the * and the ? wildcard characters work here as you would expect them to.

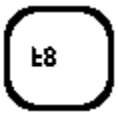


Original Path

Returns you to the original path where CT-Shell was first started.

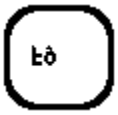
Reload Menu

Reloads the menu. After you edit modifications into your CTSHELL.INI file, you can simply press <F7> to load the new version.



Print File(s)

Will print a formatted and line-numbered listing of the current or tagged files.



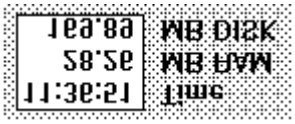
and

Are reserved for the user. These are configured by entries in the [OPTIONS] section of the CTSHELL.INI file.

Parent

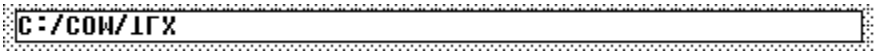
The escape key is used to change to the parent directory.

Status Display



Displays the current time, the amount of RAM that is available (including virtual memory if you're running in Enh386 mode) and how much room is left on the current disk drive. The latter two measurements are displayed in megabytes, to the nearest hundredth.

Current Path



Click the mouse on any part of the path that's displayed, and you'll change immediately to that directory. CT-Shell will stay, and continue to work, in the new directory that you've chosen. However, you are still able to press <F6> at any time to go directly to the original drive and directory where CT-Shell was started.

Files Window

The largest window contains a display of the files in the current directory. Information displayed for files includes name, extension, size in bytes, last modified date, last modified time, and attributes.

Deleting Files

If you press <Del> , the current file or an entire set of tagged files can be deleted. Additional deletion options include a CT-Shell keyword called DELDIR, which deletes an entire directory.

Name

Directory names are displayed in uppercase, to distinguish them from file names. The filename extension is included in this field. In addition, following the directory and file listings, the name field will display the various disk drives that are available on the system. If you would like to change the name of a file or a directory, you can easily do it with the CT-Shell RENAME keyword.

Size

The size in bytes of the file is shown here. You are also able to find out how many total bytes are included in a set of files that have been tagged, by using one of the special CT-Shell keywords, TAGGED. You are also able to discover how many subdirectories, files, and bytes a directory contains, using the CT-Shell DIRSIZE keyword.

Date

The date when the file was last modified (created or updated) is shown using the conventional mm/dd/yy format. Both the time and the date for a file or a group of tagged files can be changed using the CT-Shell keyword, SETDATE.

Time

CT-Shell displays the file's creation time in its full resolution, which is to within two seconds.

Attributes

The file attributes are displayed as a series of characters which may include RHSDA, for read only, hidden, system, directory, and archive, respectively.

Read/only

A file with the read/only attribute cannot be overwritten or deleted.

Hidden

Hidden means that a file won't show up in an ordinary DIR command from the DOS command processor, and the DOS COPY command won't copy a hidden file.

System

System means the file is a special type which is part of DOS itself.

Directory

Directory makes the file a subdirectory, rather than a data file or a program.

Archive

The archive attribute means that a file has been changed since the last time it was backed-up.

Changing Attributes

To alter the attributes for a single file or for a group of tagged files, execute a menu entry that uses the keyword ATTRIB.

Disk Drive Display

At the end of the files listing you'll find entries for all the disk drives in your system. Each is identified as to type and each (except floppies) shows its current remaining capacity.

Extended Selections

The file window is programmed to allow extended selections. Thus, you can tag multiple files by holding down the <Shift> or <Ctrl> keys as you tag with the mouse. <Shift> will allow you to extend a selection to include contiguous files (a group all together) and <Ctrl> will let you select any files, even if they are separated by others that you don't want tagged.

From the keyboard, press <Shift-Ctrl> as you move the arrow keys, to tag a number of files that the cursor bar passes over.

Doubleclicking Entries

What happens when you doubleclick the mouse on an entry in the files list or move the highlight bar to an entry and press <Enter> depends on what kind of file is selected:

Directories

If it is a directory, then you will change to that directory.

Executable Files

If it is an executable file (.EXE, .COM, .PIF or .BAT) you will execute that file.

Known Extensions

CT-Shell checks the [Extensions] profiles in your WIN.INI file, and can "run" files that are not themselves executable, but for which you have provided an extension in your WIN.INI file. This feature is documented in your Windows manual.

Drive Specifications

If it's one of the entries at the end of the files list that describes a disk drive in your system, doubleclicking on it will change to that drive, which will then become the default.

Command Line

To access the CT-Shell Command Line using the mouse, click on the [Command Line] button. To access the command line without using the mouse, simply press the key combination <Shift-Enter>. Here are some of the important features of the CT-Shell command line:

DOS Commands

Common DOS commands like CD, RD, MD, COPY, and DEL are handled internally in CT-Shell, without using the DOS command processor at all. Most DOS commands that CT-Shell processes are enhanced in useful ways.

CT-SHELL Commands

Some additional CT-Shell commands may be issued from this command line as well:

Find

The FIND keyword is designed to find a string (of characters) wherever it may occur within any of the files in the current directory. With the proper setup in the [EDITOR] section of your CTSHELL.INI file, you'll be able to click on a match provided by FIND and edit that file immediately.

Move

If you want to move a file quickly from one place to another, rather than copying it, you can use the CT-Shell MOVE command. The syntax is just like the ordinary COPY command, but the move is much faster.

Where

Shows you where a file is on the disk, and optionally moves to the directory where it was found. The customary DOS wildcard characters are acceptable here, so you could search for files such as:

where *.dbf or where copy??.bak

Command Recall

CT-Shell maintains an internal doubly-linked list of previous commands, and lets you scroll through them to select a command to issue again. Each command that you type at the command line is added to the list, and there are three options for deleting old commands that you no longer want to scroll through.

CTSHELL.INI Reference

The CTSHELL.INI file contains all the configuration information that makes your version of the program unique to your system and to the way you use it. There are five sections within it that control the way CT-Shell works: EDITOR, OPTIONS, COLOR, PRINTER, and ITEMS.

Editor

The first section is marked [EDITOR] and contains two settings that tell CT-Shell whether you have a text editor and whether it has a particular capability. These settings are used in conjunction with the FIND command, and will allow you to edit the file that contains a string of characters that you have asked CT-Shell to find for you. These settings look like this in the sample CTSHELL.INI file:

```
[EDITOR]
EDITORNAME=NOTEPAD.EXE
EDITLINE=
```

The first of these specifies the path for an editor. The second specifies the command line switch that can be used to specify which line in the file to edit.

Options

The [OPTIONS] section provides a place to tell CT-Shell which disk drives you want it to ignore as it creates the file listings. It can be convenient for all available drives to be shown in the listing, as their free capacity is displayed, and the user can doubleclick on their entries to change to those drives. However, each time the directory listing is refreshed, those drives are checked and their information updated if it has changed. That takes time, and if there are drives that you do not frequently change to, and don't care about their remaining capacity, you can speed up general operations considerably by telling CT-Shell to ignore them.

You do that with an IgnoreDrives= entry in the [OPTIONS] section of your CTSHELL.INI file. This example shows how you would limit the drives to just A, B, C, D and E, in a system that has many more available:

```
[OPTIONS]
IgnoreDrives=FGHIJKLMNOPQRSTUVWXYZ
```

You may change this option during a CT-Shell setting and press the <F7> function key to reload your menu and reset this option.

This section also contains two complete commands for user options that CTShell will execute when the user presses the <F9> and <F10> function keys, or clicks the mouse on their representation on the screen. The order of the fields is the same as for menu entries, which are covered in more detail in the subsection about [ITEMS].

(Note: in some versions, <F10> may be available only from the screen.)

Color

The [COLOR] section allows you to control the color of the main window background in CTShell. There are three color components that you can set individually–red, green and blue–that together make up the window background color.

Having changed these three values in your CTSHELL.INI file, you can select the menu entry that executes your COLOR keyword. Doing that re-reads the color values from your CTSHELL.INI file and resets the background color. Alternatively, you can restart CTShell, and it will use its new colors when it loads. Changing the background color is something that most people are likely to do just one time, when they first begin using CT-Shell.

Printer

The [PRINTER] section controls certain options that affect the way CT-Shell prints a file (or a list of tagged files) when you press the <F8> key. It looks like this in the sample CTSHELL.INI file:

```
[PRINTER]
LINESIZE=80
HEADINGS=1
PAGENUMS=1
24HOUR=0
LINENUMS=1
INDEPENDENT=0
DRAFT=0
TEXTFIXED=1
```

All of these settings are controlled from within CT-Shell, from a dialog box that is presented when you execute a popup menu entry that uses the CT-Shell keyword CONFIG.

Linesize

The LINESIZE entry will be 80, 110 or 132, if it was entered from within CT-Shell, and it specifies the width of text file lines, as you usually work with them. When printing text files, CT-Shell will choose the largest font that is available for your printer that will display at least that many characters on a single line, in addition to allowing room for borders and optional line numbers.

The rest of the options shown here are simply TRUE/FALSE values, where the number 1 represents TRUE and 0 represents FALSE.

Headings

HEADINGS determines whether name/date/time headings will be printed at the top of your listings.

Pagenums

PAGENUMS determines whether your listings will have page numbers at the bottom of each page.

24Hour

This option lets you determine whether the times displayed in file listings will be in 24-hour "military" time, or in the conventional AM and PM format.

Linenums

This option will cause lines to be numbered from 1 to 99999, and separated from the text with a > and a space.

Independent

The INDEPENDENT option refers to page-numbering for multiple-file printing

jobs. If you have a series of files tagged before you press <F8>, all of them will be printed, not just the current file. If INDEPENDENT is set to TRUE, each file listing will begin with page 1. If INDEPENDENT is set to FALSE, the whole series of files will be page-numbered consecutively, straight through.

DRAFT

DRAFT tells Windows whether to try to find a font for high-quality output or one that will print more quickly, with lower quality. If DRAFT is TRUE, lower quality will be allowed.

TEXTFIXED

Whether the text portion of your printout is printed in a fixed font or a variable (proportional) font is controlled by this one.

Items

This part of your CTSHELL.INI file determines the contents for your menu, and the popup menus that its items invoke.

Menu Items

Each menu item is distinguished by the special word "Item" that appears first on its line, then a set of braces containing the item name as it should appear in the main menu.

Item Name

The menu item name may contain embedded spaces, and it may contain the special ampersand character (&), which determines a letter that will appear underlined in the menu itself. The following example defines a menu item named *File:*

item {&File}

Popup Entries

Each popup entry contains five fields, delimited by braces. These fields determine what action is to be taken when the user selects that entry from a popup menu:

{*EntryName*} {*DirPath*} {*ExePath*} {*Switches*} {*Keyword*}

Entry Name

The entry name is displayed in the popup menu to allow the selection of this option. Like the menu item name, the entry name may contain an ampersand character to determine the character that will be underlined in the menu.

Two other special characters may be used here to provide separation of menu items. They have been chosen because they give somewhat the appearance of what they do in a menu. The entry name may begin with a hyphen (-), such as:

{-E&xit Program}

to draw a horizontal line between the previous menu entry and this one, and it may be preceded by a plus sign (+) to cause the entry to begin a new column within the popup menu, with a vertical bar separating the two columns:

{+&Edit SYSTEM.INI File}

Only one of these characters may be used at a time, although it may be combined with the ampersand, as in the second example above. If more than one of these special characters occurs in a menu entry, each extra one will simply show up as part of the name.

Spaces may be used to indent menu entry names, however you will want to remember that proportional spaced fonts are used in the menu displays, so you may need to experiment a bit to get indentation the way you want it.

DirPath

The directory path is an optional field which, when provided, causes CT-Shell to change either temporarily or permanently to that directory.

ExePath

The executable path is the path name for an executable file which is to be run when this entry is selected.

Switches

Programs often require additional information on the command line when they are run. An editor, for example, can often be told what file to edit by including the file name as part of the editor command.

The switches field is the one in which the CT-Shell special field characters are most often used (see below). Using object-oriented techniques, you are able to include the current file as part of your command, a list of all the tagged

files, an environment variable, and more. You are even able to cause CT-Shell to prompt you for one or more values to be inserted as it runs.

Keyword

Many of the operations that CT-Shell performs are handled internally by CTShell itself. That's how it is able to improve on many of the DOS commands rather than passing the commands along to DOS. A keyword command is always used in place of–rather than in addition to–an executable path. The following example implements the ABOUT keyword in a menu entry that is named *About CT-Shell:*

```
{&About CT-Shell} { } { } { } {about}
```

CT-Shell Keywords

About

Displays information about CT-Shell, including the number of the version that you're using.

Attrib

Changes file attributes. Use this keyword in a menu item to let you change the attributes of the current file.

Color

Resets the background color based on three color values stored in your CTSHELL.INI file.

Command

Invokes the command processor that is associated with your COMSPEC environment variable.

Config

Configures the settings within CT-Shell that affect how file listings are printed.

Copy

Copies a file to another location.

Deldir

Deletes the currently-selected directory and all files in it.

Delete

Deletes a file.

DirSize

Displays a listing that shows you how big a directory is.

Exit

Shuts down CT-Shell.

Help

Runs the Windows help engine. This provides access to the online helpfile that you are now reading.

Home

Changes the CT-Shell "home" directory to the current directory, making this the place where you'll return when you press the <F6> key, rather than the directory in which CTShell was started.

Move

Moves a file to another location. This feature changes the directory information relative to a file without copying the file itself.

Printer

Invokes your printer driver setup function.

Rename

Changes the name of a file or directory.

System

Displays the same information you get from the Windows Program Manager by clicking on its HELP/ABOUT option.

Tagged

Displays the number and size of tagged files.

Touch

Changes the date/timestamp on the current file to the current date/time.

(The keywords from this point on are designed to work with sets of tagged files, however they may be used to affect the current file only, if it is the only file that is selected.)

SetDate

Changes the date/time stamp that DOS has applied to a file or a set of files.

Tattrib

Changes the attributes of tagged files.

Tcopy

Copies a set of files to another location.

Tdelete

Deletes a set of tagged files.

Tmove

Moves a set of tagged files to another location.

Ttouch

Changes the date/timestamp on all the tagged files to the current date/time.

Special Field Characters

Many times a command should contain the name of the current file, or a list of all the tagged files, or other additional information. Sometimes it is convenient to refer to an environment variable in the directory path field, so that the command doesn't need to be changed just because the directory has been changed. There are a number of special field characters that allow you to insert such information into a command in a convenient object-oriented manner.

The term "object-oriented" here means that you do not need to write special code using a programming language, or execute a function or procedure to do these things. Certain objects (characters like ! and #) that you place in the command automatically take on values that represent file names or other information.

Here is a listing of all the special field characters that may be used in CTShell popup menu entries. Although any of them may be used in any of the fields except the entry name field and the keyword field, you will find that certain ones are likely to be used in the directory path field, and other ones are more likely to be of use in the executable path and switches fields.

!

The exclamation mark translates into the current file name. The following example uses the Windows NOTEPAD editor to modify the current file:

```
{&Edit} { } {notepad.exe} {!} {}
```

#

The hashmark translates into a list of files that are tagged, or as many of them as can be squeezed into the DOS limit of 127 characters on a command line. The following example uses the popular LHArc program to create an archive named ARCNAME.LZH that contains the files that have been tagged:

```
{LHArc &Add} { } {lha.exe} {a ARCNAME #} { }
```

??

A pair of question marks surrounds the prompt you want CT-Shell to display when it asks you for a string of characters to put in its place. This is how you can supply variable arguments at the time an entry is executed. The following example uses the popular LHArc program to create an archive of a name that is supplied by the user, that contains the files that have been tagged:

```
{LHArc &Add} { } {lha.exe} {a ?Archive name? #} { }
```

%%

A pair of percent signs will cause CT-Shell to insert the value for a named

environment variable into the command. This is consistent with the way environment variables can be accessed within a batch file, and the topic of environment variables is explained fully in your DOS manual. When used in a CT-Shell popup entry, the two percent signs and the variable name that is between them are replaced by the value that DOS associates with that variable. The following example adds the current file (assumed to be an object module) to the library named FOO.LIB in the directory that is named in the environment variable LIB:

```
{Add &Module} { } {lib.exe} {%LIB%\FOO +!;} { }
```

In addition to environment variables that are defined by SET commands issued to DOS, CT-Shell recognizes a special pseudo-environment-variable called WINDIR that represents the Windows "home" directory. CT-Shell knows where this is without being told, and will insert that path wherever you use the expression %WINDIR% in your menu entries. It can be used anywhere an ordinary environment variable can. The following example uses the WINDIR pseudo-environment-variable in an entry that edits the WIN.INI file, which is stored in the Windows directory:

```
{Edit &WIN.INI} { } {notepad.exe} {%WINDIR%\win.ini} { }
```