NAME

ded – directory editor

SYNOPSIS

ded *[options] [file-specifications]*

DESCRIPTION

The directory editor provides the user with a full-screen interface to POSIX files and directories. Multiple lists of files may be displayed, sorted by various fields, and used in built-in or ordinary shell commands. Navigation among directories is either hierarchical, or by means of a directory tree screen.

Ded provides you with a **vi**-like interface to files and directories. Two types of displays are provided:

- file list. One or more files are displayed, one per line. The current working directory is shown at the top of the screen. The bottom of the screen (marked with a dividing line) is a work area in which shell commands may be issued, and their results displayed. You may move the cursor forward or backward through a file list, sort the list, and apply various commands to the files shown.
- directory tree. One or more lines are shown, each representing a directory (or symbolic link to a directory). You may move the cursor about the directory tree both by arrow keys, as well as by issuing commands to search for particular names.

Ded manages multiple file lists; commands are provided for opening new lists from either a file list or from the directory tree. You may page between file lists, or back and forth to the directory tree.

Within a file list, you may mark groups of files on which subsequent commands will operate. Both built-in and shell commands operate upon the marked files, and may be repeated for different files.

Ded uses the *curses (3)* screen management package and runs in a POSIX environment.

OPTIONS

Most command-line options initialize the state for display-manipulation commands. They include:

-a show "." and ".." entries in each file list (see "**&**" toggle).

-b use the box-character set for displaying the directory-tree and workspace ruler.

-c *filename*
: read user commands from the specified file. The commands are assumed to be in the same format as the log-file generated with the "**-l**" option. Subprocesses do not (currently) inherit this option.

-D force date-display to long format. Allows editing of modification times using the "T" command.

-d enable debug trace.

If the standard output is a terminal, debug messages are written to the standard output. Otherwise, they are written to the standard error.

Repeating the flag changes the level. Setting the **DED_DEBUG** environment to a nonzero integer will also enable a debug-trace, with the corresponding level. The levels are cumulative:

1 **ded** prints the name of each file as it gets its properties, and pauses after reading the set of files.

2 In addition, **ded** prints the data read from a pipe.

3 On exit due to error, **ded** will dump core.

-e modify the interpretation of the "e" command to spawn a new process of **ded** when editing a directory.

-G toggle user/group display field (initially displays user-identifiers).

-I toggle inode/links display field (initially displays number of links for each file).

-i invert colors used for filenames. This helps with contrast on white backgrounds, particularly with older curses implementations which do not support default-colors.

-l *filename*
: log user commands (and **ded** messages) to the specified file. **Ded** subprocesses invoked with the "e" command inherit this option.

-m disable mouse events.

-n disable prompt on **ded** quit command.

-P toggle file-protection display field (initially displays file protection in **ls**-style).

-p print full pathnames of tagged files on exit.

-O show rcs file-lock owner.

-r*key*
provide initial reverse-sort by *key*

-S toggle file size (in blocks) display field (normally displays only the size in bytes).

-s*key*
provide initial sort by *key*

-T toggle date-display to long format.

-t*path*
read the directory tree file ".ftree" from the specified directory *path*. This option is inherited in subsequent processes of **ded**.

-x toggle BRE (basic regular expressions) versus ERE (extended regular expression).

-V show rcs/sccs version.

-Z toggle rcs/sccs data display (normally inactive).

-z toggle rcs/sccs data display

OPERATIONS

Most **ded** commands are single-character, like those of **vi**. Where appropriate, **ded** commands may be prefixed with a repeat *count*. (When not specified, the repeat *count* is always one).

Invoking DED

When you invoke **ded**, it scans its argument list. Options must precede file specifications. Arguments which do not begin with "-" are treated as file specifications. File specification arguments may begin with "~" (tilde) to denote C-shell style home-directory specifications.

You may redirect **ded**'s standard-input to a list of file-specifications (e.g., the output of **find**).

If only one file specification is given, **ded** checks to see if it is the name of a directory. If so, **ded** changes its working directory to that, and shows all files which reside in the directory. If any files are found, **ded** builds a display showing the files which it found, in an **ls**-like scrollable display. If no arguments are given, **ded** assumes the current working directory, ".".

If more than one file specification argument is given, **ded** changes its working directory to the most common pathname among the arguments. Note that if you give **ded** a wildcard argument such as "d*", then the **shell** performs wildcard expansion, not **ded**.

Items from the argument list which are not found are not displayed (though they are retained in the argument list for subsequent rescans with the "**R**" command). Arguments may be given in absolute or path-relative names (i.e., beginning with "/" or some sort of ".." construct).

Exiting From DED

To exit from **ded**, you must be in the file list display. If you are in the directory tree, typing "**q**" (for **quit**) will release the current file list and move to the next file list (if any). Typing "**Q**" moves backward. When only one file list is left, **ded** will toggle back to the file list display. If there are a lot of file lists, it is faster to toggle back directly with "**D**". (A repeat *count* may be applied to the directory tree quit-command).

To exit from **ded's** file list display, type "**q**" (for **quit**). If you have entered any other file lists (or a subprocess of **ded**), you will be prompted to ensure that you really intend to quit the current process. (If you specified the "**-n**" option on invoking **ded**, this prompt is bypassed).

Interrupting DED

You may also forcibly exit from **ded** by typing your terminal's quit character. This causes **ded** to exit without updating the ".ftree" file (the directory-tree database).

Some of **ded**'s commands may take a long while to execute. You may interrupt these commands by typing your terminal's interrupt character:

- The directory-tree "**R**" command may be interrupted between individual files in the directory-scan.
- inline changes which propagate to a tagged group of files (i.e., the "**p**", "**u**", "**g**" or "**=**" commands) may be interrupted between individual files in the group.
- the "**t**" type-file command.

Cursor Movement and Scrolling

Cursor movement in **ded** is styled after **vi** (i.e., the h, j, k and l keys). However, since there are two types of displays, there are some differences:

- In file list displays, you may normally move the cursor only up and down. This frees the left/right keys for other uses. As you move the cursor up and down in the display, it stays in the column immediately before the file names. The *left-arrow* and *right-arrow* keys scroll the screen left and right, respectively.
- Movement in the directory tree is two-dimensional. You may move the cursor left or right (up or down levels of the directory tree) or up or down (to different directory names). The screen display of the directory tree has markers (vertical bars or dashes) showing where you are allowed to put the cursor. Vertical movement is normally within items at the same directory level. You may move from one line to another at any level by the "**J**" and "**K**" commands.

Single-character cursor movement commands are:

h (directory): same as *left-arrow*.

k move cursor up *count* entries (also: *backspace*, *up-arrow*).

j move cursor down *count* entries (also: *return*, *down-arrow*).

l (directory): same as *right-arrow*.

f scroll forward *count* screen(s).

b scroll backward *count* screen(s).

left-arrow
(file list): scroll left *count*/4 screen(s).
(directory): move up *count* directory level(s).

right-arrow
(file list): scroll right *count*/4 screens.
(directory): move down *count* directory level(s). The cursor is limited by the rightmost name on the current line.

J (directory): move down *count* lines(s).

K (directory): move up *count* line(s).

H moves cursor to the first entry on screen.

M moves cursor to the middle of screen.

L moves cursor to the last entry on screen.

^ repositions the screen with current line at the top. If the current line is already at the top, **ded** toggles, putting it at the bottom of the screen.

Within either the file list or directory tree displays, you may scroll to different items in the ring of file lists. Within a file list, the ring-scrolling pages to a different file list display. Within the directory tree, the ring-scrolling simply moves the cursor (and changes the context marker) to the specified file list.

Ring-scrolling commands are:

F scroll forward (through the ring of file lists) *count* entries.

B scroll backward (through the ring of file lists) *count* entries.

Search Commands

You may move the cursor by searching for a particular string. The following search commands are provided a la **vi**:

/ **Ded** will prompt you for a regular expression. If you give it one, it will search forward (with wrap-around) for it. A return without text will cancel the search.

? **Ded** prompts you for a target and searches backwards (with wrap-around) for it.

n continue previous search (in the prevailing direction).

N continue the previous search, but in the reverse direction.

Ded maintains a separate search context for the file list and the directory tree display. That is, the targets are maintained separately. Searches in the directory tree do not include the "/" marks which separate path names; you may search only for the leaf names.

Display-Adjustment

You may use the following file-list commands to alter the format of the display, to refresh it, or to re-stat specific lines. Several commands are provided for toggling the display format:

& toggles display showing "." and ".." entries in each file list.

CTL/G

toggles a status display in the header which shows the number of files tagged and their total size (in blocks). Type "2*CTL/G*" to show the total size in bytes.

@ toggle the display of symbolic links. When active, **ded** displays the mode, owner and group of the target of the link, rather than the link itself. To make this simple to see, **ded** displays the mode in uppercase. If you apply an inline command (i.e., "**p**", "**u**" or "**g**") to a tagged group containing a symbolic link, **ded** will automatically toggle the display to display the targets rather than the links.

C toggle date-field to display. POSIX maintains three file dates (**c**hanged, **m**odified and **a**ccessed). The current state of this toggle is shown in the screen heading (e.g, "**[mtime]**").

G toggle user/group display field. Type "2G" to show user and group names at the same time.

I toggle inode/links display field. Type "2I" to show inode and device code at the same time.

P toggle file protection-mode (octal/normal) display field. This is useful because occasionally the POSIX protection display is ambiguous. Type "2P" to show the user/group field's numeric value as well.

S toggle the display of file size between bytes and blocks. Type "2S" to show both fields at one time.

T toggles the display of file-dates, or allows you to edit the modification time if the "-D" option was given.

Normally **ded** displays the file-dates in a compact form based on the relative dates. The long form shows all information returned by *ctime (2)*. Type "2T" to show a number which represents the age of the files in days (and fractions thereof). Type "3T" to show the file timestamp in seconds, e.g., since 1 Janaury 1970.

See the section *Built-in Operations on Groups of Files* for details of editing the modification time.

X toggles the screen between one and two viewports. You may adjust the size of these viewports with the "**A**" and "**a**" commands. The two viewports share the same scrolling sense, but have an independent notion of the current file.

Other commands (which do not simply toggle between different displays) are:

tab moves the cursor to the other viewport.

A move workspace marker up *count* line(s).

a move workspace marker down *count* line(s).

CTL/R
: causes **ded** to prompt you for a regular expression (see **ex**(1)) which will control the set of files subsequently shown in the current file-list. **Ded** then rescans the current directory, adjusting the file-list.

R re-scan argument list (refresh). This clears file grouping, re-reads all of the directories specified in the argument list, re-sorts and repaints the screen. The cursor is moved to the first file in the list. (If nothing is found, **ded** will exit).

Directory names which are encountered in the scan are added to the directory-tree. If the "@" toggle is set (see above), symbolic links which resolve to actual directories are also added.

W re-stat entries in the current screen. If a repeat count is given, this command is repeated at 3-second intervals (or until interrupted).

w refresh (i.e., repaint) the window.

l re-stat the current entry, as well as files which are grouped. If a repeat count is given, this command is repeated at 3-second intervals (or until interrupted).

space
: clear workspace window. This command is particularly useful after executing a shell command, since *curses* has no notion of what is shown in the workspace.

CTL/K
: causes a copy of your current screen (obtained from the curses window-state) to be appended to the file "ded.log" in your process's home directory.

Sorting the Display

You may issue commands for sorting the file-list. The cursor remains positioned at the same file after a sort. (The directory tree is always sorted alphabetically).

r*key*
: reverse-sort entries.

s*key*
: sort entries in the "normal" order (dates and other numeric fields in descending order, names in ascending order).

The *key* suffix denotes the fields which are compared, and are always a single character:

@ sort by symbolic-link target-names

. sort, using "." characters as column-separators.

c last-change date (e.g., chmod)

d directory-order (i.e., order in which items were read from the directory)

D device-code (displayed when you toggle the display with "2I").

g group-identifier (lexically)

G group-identifier (numerically)

i inode

l number of links

n filename

N filename leaf (i.e., without directory names)

o rcs/sccs lock-owner

p file-protection mask/mode.

r last-access (read) date

s file size (bytes)

S file size (blocks)

t file type (after first ".")

T file type (after last ".")

u user-identifier (lexically)

U user-identifier (numerically)

v rcs/sccs versions

w last-modification (write) date

z rcs/sccs check-in dates

Z rcs/sccs check-in dates over modification dates

You may sort tagged files separately from the remainder of the files by following the "**r**" or "**s**" character with a "**+**". For example, "s+w" sorts the file list into two parts, with the tagged files at the beginning, and untagged files following (both lists sorted by modification date).

To make sorting simpler (there are, after all, a lot of possibilities), ded recognizes the following special sort-keys:

? directs **ded** to show the current sort-key as a message.

: causes **ded** to prompt you for the sort-key. In response to any legal sort-key, **ded** immediately shows the message describing the sort. You may scroll through the list of possibilities using the up/down arrow keys. To complete the selection, press *return* (or *newline*). To quit without sorting the list, type "q".

newline
(or *return*) directs **ded** to resort the file list using the current direction (i.e., "s" or "r") specifier and the last sort-key.

Grouping Items in the Display

Both the file-list and directory tree support the notion of tagging or grouping items in the display. Groups in the file list mark files upon which commands can operate as a whole. Directory tree groups are used to mark entries for purging from the database. The following commands mark and unmark items for the group:

+ Add the *count* entries to the *group*. Grouped items are highlighted in the display. **–** Remove the *count* entries from the *group*.

_ Remove all entries from the *group*.

Tags (or untags) all files which are currently sorted adjacent in the file list, which have the same sort-key. For instance, if the list is sorted by modification date, the "**#**" command tags all files which have the same modification date as the current entry. To tag all files having the same sort-key as a neighbor in the file list, use "2#". To untag files having the same sort-key as the current file, use "0#".

Inline Editing of the File List

Ded provides you with several built-in commands to modify fields of the display. An inline editing command is initiated with a single character. Typing this character again (while in **cursor** mode!) toggles out of the inline editor.

Initially, the inline editor is in **cursor** mode. If you are editing a text field (i.e., user-identifier, group-identifier or file-name), then you may toggle to **insert** mode by typing *CTL/I* (*tab*).

q (cursor) aborts the command.

command
(cursor) completes the command. A *newline* or *return* in either mode will also complete it.

printing
(insert) Typing a printing character while in **insert** mode causes that character to be inserted into the field.

erase-char
(insert) deletes the character to the left of the cursor.

erase-word
(insert) deletes the word to the left of the cursor.

kill-char
(insert) deletes the character at the cursor position.

left-arrow
(cursor) moves cursor left one column within the edited field.

The *backspace* key does the same operation.

The *h* key (like *vi*) also moves the cursor left.

right-arrow
(cursor) moves cursor right one column within the edited field. The *form-feed* (*CTL/L*) key does the same thing. The *l* key (like *vi*) also moves the cursor right.

up-arrow
saves the current set of editing keystrokes and replays an older set from the command's history.

CTL/P does the same thing.

down-arrow
saves the current set of editing keystrokes and replays an newer set from the command's history.

CTL/N does the same thing.

CTL/I toggles between **cursor** and **insert** modes. While in **cursor** mode for text fields, the field is highlighted and prefixed with a "^" character.

CTL/B move cursor to beginning of field

The ^ key (like *vi*) also moves the cursor to the beginning.

CTL/F move cursor to end of field

The *$* key (like *vi*) also moves the cursor to the end.

b, *w*, and *e*
(cursor) move the cursor to the beginning of the previous word, or to the next word, or to the end of the current/next word, respectively as in *vi*.

Built-in Operations on Groups of Files

Inline file-oriented commands operate on the current entry. Where appropriate, commands operate on a tagged *group* of entries as well. (If any files are grouped, the file list heading is highlighted).

p Edit protection-code for *group* of entries. The code from the current entry is edited inline. The result is used for all selected entries. Editing is done with single characters:

p complete command (a newline or return also completes the command)

q abort command

octal-digit
set chmod-field to the given value, which must be in the range 0 to 7.

space
move cursor right 3 columns (or the next octal digit if the "**P**" toggle is in effect).

backspace
move cursor left 3 columns (or to the previous octal digit if the "**P**" toggle is in effect).

P toggles display mode (current line only) between octal and normal.

s toggles "set user id" or "set group id" bit, according to the position of the cursor.

t toggles "save swapped text" bit if cursor points to the last chmod field.

CTL/B
move cursor to beginning of field

CTL/F
move cursor to end of field

u Edit user-identifier field. The current entry's *uid* is edited inline and the result used for all selected entries.

g Edit group-identifier field. The current entry's *gid* is edited inline, and the result used for all selected entries.

T If the "-D" option was given, you can edit the selected file or directory's modification time by moving the cursor left/right and using "+" or "-" to increment or decrement the part of the modification time on which the cursor lies.

= Edit name of current file. Files which are grouped are renamed using the **template** formed by your command. For example, you might rename files ending in ".o" to end in ".bin" by typing "**=** *CTL/F CTL/I erase* **.bin**".

> Edit link-text of the current symbolic link. Symbolic links which are grouped are renamed using the **template** formed by your command. For example, you might edit links beginning with "/local/bin" to begin with "/usr/local/bin" by typing "**=** *CTL/I* **/usr**".

< Like "**>**", except that the special substrings "**%F**", "**%B**", "**%D**" and "**%d**" are translated into the forward, backward, original and current directory paths in the ring of file lists, and "**#**" is translated into the current entry's name.

" Repeats the last "**p**", "**u**", "**g**", "**=**" or "**c**" command. This uses the actual set of characters typed for the command, so an editing template may be made. (**Ded** buffers the last set of editing keystrokes for inline editors).

'*xx* Repeat the last *xx* command, where *xx* is one of the inline-editing commands (e.g., "p", "u", "g", "=", etc). For example, type

```
'cf
```

to replay the last create-file command. **Ded** replays the specified command, not including the final newline. You may modify or reject the command.

Creating New Entries

You may add new entries to the display list by rescanning with the "**R**" command (to pick up new names which are added by programs other than **ded**. You may also use **ded** to create new entries using the "**c**" command:

cf create file

cd create directory

cL create hard-link (to the current file, whose name is the initial template).

cl create symbolic link (initially with "." for text).

Each "**c**" command opens the list at the current position. You must provide a name, using the same inline name-editing as the "**=**" command. When the name is complete (non-null, and non-conflicting), **ded** creates it.

The "**c**" commands may be repeated using the '"' command.

Built-in Operations on the Current File

The following built-in operations operate only on the current file, because grouping operations would not be meaningful:

E If the current entry is a directory (or a symbolic link to a directory), open a new file list on it. The new list inherits the display options and sorting sense, as well as the last shell command from the current display.

If the entry is a file, invoke the editor (e.g., **vi**) on it.

If the entry is a symbolic link to a file, **ded** opens a file list in the directory containing that file, and positions to that file.

e If the current entry is a directory, spawn a new **ded** process with that as argument. If it is a file, invoke the editor (e.g., **vi**) on it.

v If the current entry is a directory, spawn a new **ded** process with that as argument. If it is a file, invoke the browser (e.g, **view**).

m run the pager (e.g., **more**) on the current file. **Ded** will not let you page directories or other entries which are not regular files.

On return from the editor, browser and pager, **ded** prompts you (for a *return*) and then repaints the screen.

Ded provides you with a pager which operates in the workspace. For small files, or for just peeking at things, this works much faster than spawning a copy of **more**. The workspace pager displays either text or binary files:

- When displaying text files, **ded** shows sequences of consecutive blank lines as a single blank line, and shows overstruck or underlined text with highlighting. (**Ded** interprets *backspaces* and *returns* in text files). You may scroll left or right in the pager to see very long lines.
- When displaying binary files, **ded** shows control characters as ".". Non-ASCII characters (i.e., having the high-order bit set) are converted to ASCII (by stripping this bit) and shown highlighted.

As you scroll through the file, the pager shows the percentage which you have viewed a la **more**. You may use the following subcommands within the workspace pager:

q quit the pager. To prevent accidentally quitting **ded**, an immediately succeeding "**q**" command will clear the workspace.

w repaint the screen.

tab causes the tab stops used for the text-display to be toggled between 4 and 8. Use a *count* prefix to specify other tab stops.

CTL/K
causes a copy of your current screen (obtained from the curses window-state) to be appended to the file "ded.log" in your process's home directory.

A move the workspace marker up *count* lines, redisplay.

a move the workspace marker down *count* lines, redisplay.

^ scroll to the beginning of the file.

$ scroll to the end of the file.

f scroll forward *count* sub-screens (also, the *space* and *newline* keys).

b scroll backward *count* sub-screens (also, the *backspace* key).

h scroll left *count* columns (also, the *left-arrow* key).

j scroll down *count* rows (also, the *down-arrow* key).

k scroll up *count* rows (also, the *up-arrow* key).

l scroll right *count* columns (also, the *right-arrow* key).

< scroll left *count*/4 screens (also, the *CTL/L* key).

> scroll right *count*/4 screens (also, the *CTL/R* key).

The "**/**", "**?**", "**n**" and "**N**" search commands work in the workspace pager. All lines containing a match are highlighted.

The following commands use the workspace pager:

h type **ded**'s help-file in the workspace.

t type the current file, in the workspace. Sequences of blank lines are compressed to a single blank line, and overstruck text is highlighted.

To type a binary-file, use "2t". This causes **ded** to display non-ASCII bytes highlighted. Typing "3t" causes all non-ASCII bytes to be shown as blanks.

Directory-files are displayed by showing the inode and filename list via a temporary-file.

Shell commands

Shell commands are executed in the work-area. **Ded** invokes the Bourne shell via the *system (2)* call.

! Prompt for, and execute a shell command.

% Prompt for, and execute a shell command, prompting (for *return*) and repainting screen afterwards.

***** Display text of last "**!**" or "**%**" command. Use a repeat count to display items from the command stack.

: Edit text of last "**!**" or "**%**" command, re-execute.

. Re-execute last "**!**" or "**%**" command.

To re-execute a command while changing the flag which directs ded to clear the screen, use a prefix-code:

0 resets the repaint-screen flag (so that **ded** won't repaint the screen).

2 sets the repaint-screen flag.

Command Substitution

In any shell command which you issue via **ded**, you may use the special character "**#**" to cause **ded** to substitute the names of the current- and grouped-files. (A "\" preceding a "**#**" overrides this).

You may do more elaborate substitution on the current file using a two-character sequence beginning with "*%*":

%B
substitutes the name of the directory before the current one, in the ring of file lists.

%d
substitutes the name of the current directory.

%D
substitutes the name of the original directory from which **ded** was invoked.

%e substitutes the current filename, removing all but the "*.xxx*" part (i.e., "extension").

%F
substitutes the name of the following directory in the ring of file lists.

%g substitutes the group-name of the user (who owns) the current file.

%h or **%H**
substitutes the name of the current file, after removing the last component (i.e., "head").

%n or **%N**
: substitutes the name of the current file.

%o substitutes the name (if any) of the user who has reserved the current file with RCS or SCCS.

%r or **%R**
: substitutes the name of the current file, removing ".*xxx*" part (i.e., "root").

%t substitutes the current filename, removing all leading pathname components (i.e., "tail").

%u
: substitutes the name of the user (who owns) the current file.

%v substitutes the highest RCS/SCCS version of the current file, if known.

The **%N**, **%H**, **%R** and **%E** substitutions are performed after concatenating the current filename with the current directory, to make an absolute pathname.

Dollar signs and other special characters in filenames which could cause problems in command substitution are escaped (prefixed with " command.

To insert a literal "%" or "#" character, prefix it with the backslash (\) character.

Command Editing

You may edit any shell command which you issue to **ded**, either before it is issued, or after, when using the ":" command. Command editing is done in either **insert** or **cursor** modes, using the same character convention as the inline commands (see *"Inline Editing of the File List"*). **Ded** is initially in **insert** mode. When it is in **cursor** mode, the character prefixing the command-entry is set to a "^". Command editing controls are similar to the inline editor, except:

- A repeat *count* may be prefixed to any subcommand in cursor mode.
- Commands may be continued (with **ded** controlling wraparound) as long as space remains in the workspace to enter new command text.
- A *kill* character in **insert** mode aborts the command. In **cursor** mode, it deletes the *count* characters at the cursor position.

Directory Tree

Ded maintains a database of directory names. You may scroll in this display, as well as enter a new **ded** process from it. Cursor movement may be done not only up and down as in the file list, but also left and right. The "**/**", "**?**", "**n**" and "**N**" search commands work in the directory-tree (though they find only leaf names, rather than full paths).

Commands which manipulate **ded**'s file list state are:

D Toggle between directory-tree and file-list display. **Ded** will show the most recently selected file list, which is marked with "**=>**".

E Enter a new file-list at the indicated directory-name.

e Enter a new **ded** file-list with the indicated directory name. If you have specified "**−e**" on the command line, **ded** spawns a new process.

F Move forward (with wraparound) in the ring of file lists.

B Move backward (with wraparound) in the ring of file lists.

W writes the database file (if changes have occurred).

Commands which modify the display characteristics are:

& Sets a flag which causes **ded** to suppress names (and their dependents) which begin with "." or "$".

I Sets a runtime flag which disables searches into subtrees which are made invisible with "V".

V Sets a flag in the database for the current entry which directs **ded** to suppress subdirectories from the display. If you supply a repeat-count, **ded** shows up to that many levels.

w Repaint the display.

Z Directs **ded** to suppress from the display all RCS and SCCS directories.

Commands which operate upon the directory database are:

R Read directory names at the current position (also done automatically whenever a file list is constructed). If you supply a repeat-count, **ded** recurs that many levels.

Unlike the "R" command in the file-list display, this command always attempts to resolve symbolic links to directories.

+ Mark directory name for removal from database. **–** Unmark directory name.

_ Clear list of marked names.

p Purge marked names from the database.

@ **Ded** moves your cursor to the header. You may edit the path name, causing **ded** to jump to the newly specified path. The path name need not be present in the directory tree; if it is not, it will be entered into the tree.

˜ Like the "@" command, this is used to reposition the cursor within the tree. Instead of editing the current path name, you are given the home directory token "˜".

: Finally, you may position your cursor to an entry by specifying its number (displayed in the left column) by typing ":" (which causes **ded** to prompt for the number).

RCS and SCCS Commands

Ded provides you with a visual interface to *rcs* (**r**evision **c**ontrol **s**ystem) and *sccs* (**s**ource **c**ode **c**ontrol **s**ystem) files.

- For a given file, the corresponding *rcs* files (by convention) reside in a subdirectory called "RCS". The *rcs* file names are formed by suffixing the given file name with two characters (i.e., ",v").
- For a given file, the corresponding *sccs* files (by convention) reside in a subdirectory called "SCCS". The sccs file names are formed by prefixing the given file name with two characters (e.g., "p." and "s.").

Ded assumes that the *rcs* files are checked in using the script **rcsput**, and that the *sccs* files are checked in using the script **sccsput**. These scripts extend the basic *rcs* and *sccs* scheme by making the file's date and the archival check-in date the same. When directed to do so, **ded** will scan the archived files to obtain and display the most recent check-in date and version. A special display column shows the result of the comparison between the file's modification and check-in dates:

blank
no corresponding archive file was found.

= the check-in and modification dates match.

< the file's modification date is later than the check-in date.

> the file's modification date is earlier than the check-in date.

Using **ded**, you can quickly verify which files have been checked into *rcs* or *sccs*. **Ded**'s sorting options (i.e., the "**v**", "**y**", "**z**", and "**Z**" keys) facilitate this also.

The following file list commands are used for archive display:

O toggle display showing the owner of the current lock on the file. **Ded** examines the *rcs* archive file to see if there are any locks on it. If so, it displays the name of the first lock-owner

V toggle version display.

Z toggle check-in date display. The date display has three states: off, invisible (except for the comparison column), and visible. If the archive display is initially off, **ded** must scan all of the files in the current directory to see which have a corresponding *rcs* ",v" or *sccs* "s." file, and then to extract the check-in date and version number.

z clears archive display. Normally the archive display is inactive, since it does slow **ded**. If you accidentally type "**z**", you can recover the data immediately with a "**Z**" command. **Ded** does not reset the archive display data until directed to do so by a re-stat command (e.g., "**R** ", "**W**" or "**l**").

Viewing the check-in date information from within an archive directory shows the comparison of the **archived** file's modification date with the corresponding file modification date. This is mostly useful for showing archived files for which there is no corresponding checked-out file.

Logfile Format

The log file created with the "**–l**" option logs all **ded** commands. Logged commands begin with the repeat count in column one. Multi-character commands are logged on a single line, e.g.,

```
1st
1%ls -l #
```

Comments are inserted with a tab followed by a "#" character. **Ded**'s log comments indicate the names of files affected by commands, current working directory, etc., e.g.,

```
        # process 1417 begun at Thu Mar 16 09:51:11 1989
        # argv[0] = 'ded'
        # argv[1] = '-lz'
1D      # path: //dickey/local/dickey
1\r     # path: //dickey/local/dickey/bin
1E      # chdir //dickey/local/dickey/bin
1/SCCS
        # "SCCS"
1e      # "SCCS"
        # process 1631 begun at Thu Mar 16 09:51:43 1989
        # argv[0] = '//dickey/local/dickey/bin/ded'
        # argv[1] = '-l//dickey/local/dickey/z'
        # argv[2] = 'SCCS'
1+      # "s.Makefile"
1+      # "s.args.c"
1+      # "s.keycode.c"
1%ls -l #
        # execute ls -l s.Makefile s.args.c s.keycode.c
\r      # Hit <RETURN> to continue
        # elapsed time = 9 seconds
1q      # process 1631 ended at Thu Mar 16 09:52:24 1989
        # process 1417 resuming
1q      # process 1417 ended at Thu Mar 16 09:52:41 1989
```

Ded commands which are read in "raw" (single-character) mode are logged as backslash-codes, if necessary, to make them readable (e.g., "**\t**" for tab). In addition to the standard backslash codes defined for the C language, **ded** also uses

\s for space (to make it visible in the log)

\U up arrow

\D down arrow

\L left arrow

\R right arrow

\F control/F

\B control/B

\B control/W – usually word-erase

Other text (which is buffered) contains no non-printing characters.

X Windows Enhancements

Ded assumes that you are running under the X Window System. In this case, if the program **xterm** is found in your execution path, **ded** will permit the following commands:

CTL/E
: edit the current file (using the default editor invoked by the "**e**" command) in an **xterm** process. **Ded** waits until you have exited from the process before continuing.

CTL/V
: view the current file (using the default browser invoked by the "**v**" command) in an **xterm** process. **Ded** does not wait for you to exit from this process; it will proceed to accept new commands.

Ded also allows you to move the cursor by clicking with the mouse. Double-click to edit a selected item.

Ded can also handle window-resizing events, when properly configured (e.g., with BSD4.x curses, or ncurses). When running in an xterm or similar terminal emulator which supports the SIGWINCH signal, **ded** resizes the curses display structures.

Color Displays

If the curses libraries which you use to build **ded** support color (e.g., SystemV or ncurses), **ded** can display filenames in color. It is designed to use the */etc/DIR_COLORS* file which supports the Linux color **ls** program.

Ded looks for the color-file in one of the following locations:

~/.ded_colors
~/.dir_colors
/etc/DIR_COLORS
datadir/ccodes.rc
datadir/cnames.rc

The color-file specifies terminal types that can display in color, and patterns and their corresponding colors. On Linux, the colors may be specified by a series of numbers. On all systems, **ded** recognizes assignments of the form

f=white
b=blue

to specify the foreground and background of characters.

Ded does not attempt to manage the background color of your display, because that does not work well when mixed with interactive shell commands whose output goes to the workspace.

ENVIRONMENT

Ded uses the following environment variables:

PATH
: used to establish where **ded** is run from, so that the help file can be found.

DED_CM_LOOKUP
: specifies the order to use when looking for the last version of files in RCS, SCCS, CmVision, CVS. If you do not specify it, the default order is "rcs,sccs,cvs,svn", according to their availability at build time. Specify CmVision with a "cmv" keyword. Specify CVS with a "cvs" keyword. Specify SVN with a "svn" keyword.

DED_TREE
overrides the default location of the directory-tree database file. Use this to maintain separate database files on a system which has your home directory mounted on several hosts.

EDITOR
overrides default editor invoked by "**e**" command (**vi**)

BROWSE
overrides default browser invoked by "**v** " command (view).

PAGER
overrides default pager invoked by "**m**" command (more).

TERM
used to determine control sequences for cursor keys on computer systems which do not support this in *curses (3x)*.

RCS_DIR
gives the name of the *rcs* directories **ded** searches for the file list "**V**", "**Y**" and "**Z**" commands. If not specified, **ded** assumes "RCS".

SCCS_DIR
gives the name of the *sccs* directories **ded** searches for the file list "**V**", "**Y**" and "**Z**" commands. If not specified, **ded** assumes "SCCS".

FILES

When executed, **ded** determines (by inspecting the zeroth argument passed to it by the shell, as well as the contents of the **PATH** variable) where it was executed from. Its help file *ded.hlp* is checked in the *datadir* (compile-time data directory), If not found there, the help file is assumed to reside in the same directory.

The directory tree manager maintains its database in your home directory (i.e., the path is derived from your process's uid). The name of the file is "*.ftree*". If changes have been made to the memory copy of the database, this file is updated whenever **ded** spawns a copy of itself, or when exiting from **ded**.

ANTICIPATED CHANGES

Make spawned **ded** processes inherit display options from the current one.

Provide more transparent use of symbolic links (in the directory tree), including storing and showing link text.

Enhance the treatment of multiple viewports. This would permit the user to group files in one file list and then move the cursor to another file list to operate upon the group files (e.g., a bulk move without typing a path name). Additionally, the user would be able to sort the viewports independently, as well as operate upon different directories (from the directory-ring).

Use the **SHELL** environment variable, and parse arguments so that shell commands need not use the Bourne shell.

AUTHORS

Thomas E. Dickey <dickey@invisible-island.net>.

SEE ALSO

rcsput(1), **rcsget**(1), **sccsput**(1), **sccsget**(1)