NAME

exit_curses, **exit_terminfo** **-** check for memory leaks in *curses*

SYNOPSIS

#include <curses.h>
void exit_curses(int *code***);**

#include <term.h>
void exit_terminfo(int *code***);**

/ deprecated (intentionally not declared in curses.h or term.h) */*
void _nc_freeall(void);
void _nc_free_and_exit(int *code***);**
void _nc_free_tinfo(int *code***);**

DESCRIPTION

These functions are used to simplify analysis of memory leaks in the *ncurses* library.

Any implementation of curses must not free the memory associated with a screen, since (even after calling **endwin**(3X)), it must be available for use in the next call to **refresh**(3X). There are also chunks of memory held for performance reasons. That makes it hard to analyze curses applications for memory leaks. When using the specially configured debugging version of the *ncurses* library, applications can call functions which free those chunks of memory, simplifying the process of memory-leak checking.

Some of the functions are named with a "_nc_" prefix because they are not intended for use in the non-debugging library:

_nc_freeall

This frees (almost) all of the memory allocated by *ncurses*.

_nc_free_and_exit

This frees the memory allocated by *ncurses* (like **_nc_freeall**), and exits the program. It is preferred over **_nc_freeall** since some of that memory may be required to keep the application running. Simply exiting (with the given exit-code) is safer.

_nc_free_tinfo

Use this function if only the low-level terminfo functions (and corresponding library) are used. Like **_nc_free_and_exit**, it exits the program after freeing memory.

The functions prefixed "_nc" are normally not available; they must be configured into the library at build time using the **--disable-leaks** option. That compiles-in code that frees memory that normally would not be freed.

The **exit_curses** and **exit_terminfo** functions call **_nc_free_and_exit** and **_nc_free_tinfo** if the library is configured to support memory-leak checking. If the library is not configured to support memory-leak checking, they simply call **exit**.

RETURN VALUE

These functions do not return a value.

PORTABILITY

These functions are not part of X/Open Curses; nor do other implementations of curses provide a similar feature.

In any implementation of X/Open Curses, an application can free part of the memory allocated by curses:

- The portable part of **exit_curses** can be freed using **delscreen**, passing the *SCREEN* pointer returned by **newterm**.

  In some implementations, there is a global variable **sp** which could be used, e.g., if the screen were only initialized using **initscr**.

- The portable part of **exit_terminfo** can be freed using **del_curterm**.

  In this case, there is a global variable **cur_term** which can be used as parameter.

SEE ALSO

curses(3X), **curs_initscr**(3X), **curs_terminfo**(3X)