

NAME

initscr, **newterm**, **endwin**, **isendwin**, **set_term**, **delscreen** - **curses** screen initialization and manipulation routines

SYNOPSIS

```
#include <curses.h>
```

```
WINDOW *initscr(void);
```

```
int endwin(void);
```

```
bool isendwin(void);
```

```
SCREEN *newterm(const char *type, FILE *outfd, FILE *infd);
```

```
SCREEN *set_term(SCREEN *new);
```

```
void delscreen(SCREEN* sp);
```

DESCRIPTION**initscr**

initscr is normally the first **curses** routine to call when initializing a program. A few special routines sometimes need to be called before it; these are **slk_init**(3X), **filter**, **ripoffline**, **use_env**. For multiple-terminal applications, **newterm** may be called before **initscr**.

The **initscr** code determines the terminal type and initializes all **curses** data structures. **initscr** also causes the first call to **refresh**(3X) to clear the screen. If errors occur, **initscr** writes an appropriate error message to standard error and exits; otherwise, a pointer is returned to **stdscr**.

newterm

A program that outputs to more than one terminal should use the **newterm** routine for each terminal instead of **initscr**. A program that needs to inspect capabilities, so it can continue to run in a line-oriented mode if the terminal cannot support a screen-oriented program, would also use **newterm**. The routine **newterm** should be called once for each terminal. It returns a variable of type **SCREEN** * which should be saved as a reference to that terminal. **newterm**'s arguments are

- ⊕ the *type* of the terminal to be used in place of **\$TERM**,
- ⊕ a file pointer for output to the terminal, and
- ⊕ another file pointer for input from the terminal

If the *type* parameter is **NULL**, **\$TERM** will be used.

endwin

The program must also call **endwin** for each terminal being used before exiting from **curses**. If **newterm** is called more than once for the same terminal, the first terminal referred to must be the last one for which **endwin** is called.

A program should always call **endwin** before exiting or escaping from **curses** mode temporarily. This routine

- ⊕ resets colors to correspond with the default color pair 0,
- ⊕ moves the cursor to the lower left-hand corner of the screen,
- ⊕ clears the remainder of the line so that it uses the default colors,
- ⊕ sets the cursor to normal visibility (see **curs_set(3X)**),
- ⊕ stops cursor-addressing mode using the *exit_ca_mode* terminal capability,
- ⊕ restores tty modes (see **reset_shell_mode(3X)**).

Calling **refresh(3X)** or **doupdate(3X)** after a temporary escape causes the program to resume visual mode.

isendwin

The **isendwin** routine returns **TRUE** if **endwin** has been called without any subsequent calls to **wrefresh**, and **FALSE** otherwise.

set_term

The **set_term** routine is used to switch between different terminals. The screen reference **new** becomes the new current terminal. The previous terminal is returned by the routine. This is the only routine which manipulates **SCREEN** pointers; all other routines affect only the current terminal.

delscreen

The **delscreen** routine frees storage associated with the **SCREEN** data structure. The **endwin** routine does not do this, so **delscreen** should be called after **endwin** if a particular **SCREEN** is no longer needed.

RETURN VALUE

endwin returns the integer **ERR** upon failure and **OK** upon successful completion.

Routines that return pointers always return **NULL** on error.

X/Open defines no error conditions. In this implementation

- ⊕ **endwin** returns an error if the terminal was not initialized.
- ⊕ **newterm** returns an error if it cannot allocate the data structures for the screen, or for the top-level windows within the screen, i.e., **curscr**, **newscr**, or **stdscr**.
- ⊕ **set_term** returns no error.

PORTABILITY

These functions were described in the XSI Curses standard, Issue 4. As of 2015, the current document is X/Open Curses, Issue 7.

Differences

X/Open specifies that portable applications must not call **initscr** more than once:

- ⊕ The portable way to use **initscr** is once only, using **refresh** (see `curs_refresh(3X)`) to restore the screen after **endwin**.
- ⊕ This implementation allows using **initscr** after **endwin**.

Old versions of curses, e.g., BSD 4.4, may have returned a null pointer from **initscr** when an error is detected, rather than exiting. It is safe but redundant to check the return value of **initscr** in XSI Curses.

Unset TERM Variable

If the TERM variable is missing or empty, **initscr** uses the value "unknown", which normally corresponds to a terminal entry with the *generic* (*gn*) capability. Generic entries are detected by **setupterm** (see `curs_terminfo(3X)`) and cannot be used for full-screen operation. Other implementations may handle a missing/empty TERM variable differently.

Signal Handlers

Quoting from X/Open Curses, section 3.1.1:

*Curses implementations may provide for special handling of the **SIGINT**, **SIGQUIT** and **SIGTSTP** signals if their disposition is **SIG_DFL** at the time **initscr** is called ...*

Any special handling for these signals may remain in effect for the life of the process or until the process changes the disposition of the signal.

None of the Curses functions are required to be safe with respect to signals ...

This implementation establishes signal handlers during initialization, e.g., **initscr** or **newterm**. Applications which must handle these signals should set up the corresponding handlers *after* initializing the library:

SIGINT

The handler *attempts* to cleanup the screen on exit. Although it *usually* works as expected, there are limitations:

- ⊕ Walking the **SCREEN** list is unsafe, since all list management is done without any signal blocking.
- ⊕ On systems which have **REENTRANT** turned on, **set_term** uses functions which could deadlock or misbehave in other ways.
- ⊕ **endwin** calls other functions, many of which use stdio or other library functions which are clearly unsafe.

SIGTERM

This uses the same handler as **SIGINT**, with the same limitations. It is not mentioned in X/Open Curses, but is more suitable for this purpose than **SIGQUIT** (which is used in debugging).

SIGTSTP

This handles the *stop* signal, used in job control. When resuming the process, this implementation discards pending input with **flushinput** (see **curs_util(3X)**), and repaints the screen assuming that it has been completely altered. It also updates the saved terminal modes with **def_shell_mode** (see **curs_kernel(3X)**).

SIGWINCH

This handles the window-size changes which were ignored in the standardization efforts. The handler sets a (signal-safe) variable which is later tested in **wgetch** (see **curs_getch(3X)**). If **keypad** has been enabled for the corresponding window, **wgetch** returns the key symbol **KEY_RESIZE**. At the same time, **wgetch** calls **resizeterm** to adjust the standard screen **stdscr**, and update other data such as **LINES** and **COLS**.

SEE ALSO

curses(3X), **curs_kernel(3X)**, **curs_refresh(3X)**, **curs_slk(3X)**, **curs_terminfo(3X)**, **curs_util(3X)**, **curs_variables(3X)**.