NAME

panel - panel stack extension for curses

SYNOPSIS

#include <panel.h>

cc [flags] sourcefiles -lpanel -lncurses

PANEL *new_panel(WINDOW *_win_);

int bottom_panel(PANEL *_pan_);
int top_panel(PANEL *_pan_);
int show_panel(PANEL *_pan_);
void update_panels(void);
int hide_panel(PANEL *_pan_);

WINDOW *panel_window(const PANEL *_pan_);
int replace_panel(PANEL *_pan_, WINDOW *_window_);
int move_panel(PANEL *_pan_, int _starty_, int _startx_);
int panel_hidden(const PANEL *_pan_);

PANEL *panel_above(const PANEL *_pan_);
PANEL *panel_below(const PANEL *_pan_);

int set_panel_userptr(PANEL *_pan_, const void *_ptr_);
const void *panel_userptr(const PANEL *_pan_);

int del_panel(PANEL *_pan_);

/* ncurses-extensions */
PANEL *ground_panel(SCREEN *_sp_);
PANEL *ceiling_panel(SCREEN *_sp_);

DESCRIPTION

Panels are **curses**(3X) windows with the added feature of depth. Panel functions allow the use of stacked windows and ensure the proper portions of each window and the curses **stdscr** window are hidden or displayed when panels are added, moved, modified or removed. The set of currently visible panels is the stack of panels. The **stdscr** window is beneath all panels, and is not considered part of the stack.

A window is associated with every panel. The panel routines enable you to create, move, hide, and show panels, as well as position a panel at any desired location in the stack.

Panel routines are a functional layer added to **curses**(3X), make only high-level curses calls, and work anywhere terminfo curses does.

FUNCTIONS

bottom_panel

bottom_panel(*pan***)** puts panel *pan* at the bottom of all panels.

ceiling_panel

ceiling_panel(*sp***)** acts like **panel_below(NULL)**, for the given **SCREEN** *sp*.

del_panel

del_panel(*pan***)** removes the given panel *pan* from the stack and deallocates the **PANEL** structure (but not its associated window).

ground_panel

ground_panel(*sp***)** acts like **panel_above(NULL)**, for the given **SCREEN** *sp*.

hide_panel

hide_panel(*pan***)** removes the given panel *pan* from the panel stack and thus hides it from view. The **PANEL** structure is not lost, merely removed from the stack.

move_panel

move_panel(*pan***,***starty***,***startx***)** moves the given panel *pan*'s window so that its upper-left corner is at *starty*, *startx*. It does not change the position of the panel in the stack. Be sure to use this function, not **mvwin**(3X), to move a panel window.

new_panel

new_panel(*win***)** allocates a **PANEL** structure, associates it with *win*, places the panel on the top of the stack (causes it to be displayed above any other panel) and returns a pointer to the new panel.

panel_above

panel_above(*pan***)** returns a pointer to the panel above *pan*. If the panel argument is **(PANEL *)0**, it returns a pointer to the bottom panel in the stack.

panel_below

panel_below(*pan***)** returns a pointer to the panel just below *pan*. If the panel argument is **(PANEL *)0**, it returns a pointer to the top panel in the stack.

panel_hidden

panel_hidden(*pan***)** returns **TRUE** if the panel *pan* is in the panel stack, **FALSE** if it is not. If the panel is a null pointer, return **ERR**.

panel_userptr

panel_userptr(*pan***)** returns the user pointer for a given panel *pan*.

panel_window

panel_window(*pan***)** returns a pointer to the window of the given panel *pan*.

replace_panel

replace_panel(*pan***,***window***)** replaces the current window of panel *pan* with *window* This is useful, for example if you want to resize a panel. In **ncurses**, you can call **replace_panel** to resize a panel using a window resized with **wresize**(3X). It does not change the position of the panel in the stack.

set_panel_userptr

set_panel_userptr(*pan***,***ptr***)** sets the panel's user pointer.

show_panel

show_panel(*pan***)** makes a hidden panel visible by placing it on top of the panels in the panel stack. See **COMPATIBILITY** below.

top_panel

top_panel(*pan***)** puts the given visible panel *pan* on top of all panels in the stack. See **COMPATIBILITY** below.

update_panels

update_panels() refreshes the *virtual screen* to reflect the relations between the panels in the stack, but does not call **doupdate**(3X) to refresh the *physical screen*. Use this function and not **wrefresh**(3X) or **wnoutrefresh**(3X).

update_panels may be called more than once before a call to **doupdate**, but **doupdate** is the function responsible for updating the *physical screen*.

DIAGNOSTICS

Each routine that returns a pointer returns **NULL** if an error occurs. Each routine that returns an int value returns **OK** if it executes successfully and **ERR** if not.

Except as noted, the *pan* and *window* parameters must be non-null. If those are null, an error is returned.

The **move_panel** function uses **mvwin**(3X), and will return an error if **mvwin** returns an error.

COMPATIBILITY

Reasonable care has been taken to ensure compatibility with the native panel facility introduced in System V (inspection of the SVr4 manual pages suggests the programming interface is unchanged). The **PANEL** data structures are merely similar. The programmer is cautioned not to directly use **PANEL** fields.

The functions **show_panel** and **top_panel** are identical in this implementation, and work equally well with displayed or hidden panels. In the native System V implementation, **show_panel** is intended for making a hidden panel visible (at the top of the stack) and **top_panel** is intended for making an already-visible panel move to the top of the stack. You are cautioned to use the correct function to ensure compatibility with native panel libraries.

NOTE

In your library list, libpanel.a should be before libncurses.a; that is, you should say "-lpanel -lncurses", not the other way around (which would give a link-error with static libraries).

PORTABILITY

The panel facility was documented in SVr4.2 in *Character User Interface Programming (UNIX SVR4.2)*.

It is not part of X/Open Curses.

A few implementations exist:

- Systems based on SVr4 source code, e.g., Solaris, provide this library.

- **ncurses** (since version 0.6 in 1993) and **PDCurses** (since version 2.2 in 1995) provide a panel library whose common ancestor was a public domain implementation by Warren Tucker published in *u386mon* 2.20 (1990).

  According to Tucker, the SystemV panel library was first released in SVr3.2 (1988), and his implementation helped with a port to SVr3.1 (1987).

  Several developers have improved each of these; they are no longer the same as Tucker’s implementation.

- NetBSD 8 (2018) has a panel library begun by Valery Ushakov in 2015. This is based on the AT&T documentation.

FILES

panel.h interface for the panels library

libpanel.a the panels library itself

SEE ALSO

curses(3X), **curs_variables**(3X),

This describes **ncurses** version 6.2 (patch 20210220).

AUTHOR

Originally written by Warren Tucker <wht@n4hgf.mt-park.ga.us>, primarily to assist in porting *u386mon* to systems without a native panels library.

Repackaged for ncurses by Zeyd ben-Halim.

Juergen Pfeifer and Thomas E. Dickey revised/improved the library.