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();
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)

DESCRIPTION

Panels are **ncurses**(3NCURSES) 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 **ncurses**(3NCURSES), make only high-level curses calls, and work anywhere terminfo curses does.

FUNCTIONS

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.

update_panels()
refreshes the virtual screen to reflect the relations between the panels in the stack, but does not call doupdate() to refresh the physical screen. Use this function and not **wrefresh** or **wnoutrefresh**. **update_panels** may be called more than once before a call to doupdate(), but doupdate() is the function responsible for updating the physical screen.

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

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

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

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

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

move_panel(pan,starty,startx)
moves the given panel 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()**, to move a panel window.

replace_panel(pan,window)
replaces the current window of panel with **window** (useful, for example if you want to resize a panel; if you're using **ncurses**, you can call **replace_panel** on the output of **wresize**(3NCURSES)). It does not change the position of the panel in the stack.

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(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.

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

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

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

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.

COMPATIBILITY

Reasonable care has been taken to ensure compatibility with the native panel facility introduced in SVr3.2 (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 want to say '-lpanel -lncurses', not the other way around (which would usually give a link-error).

FILES

panel.h interface for the panels library

libpanel.a the panels library itself

SEE ALSO

ncurses(3NCURSES)

This describes **ncurses** version 5.7 (patch 20100313).

AUTHOR

Originally written by Warren Tucker <wht AT n4hgf DOT mt-park DOT ga DOT us>, primarily to assist in porting u386mon to systems without a native panels library. Repackaged for ncurses by Zeyd ben-Halim.