**NAME**
   panel - panel stack extension for *curses*

**SYNOPSIS**
   **#include <panel.h>**

   **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 property of depth.  Panel functions allow the use of
   stacked windows and ensure that 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 each panel.  The panel routines enable you to create, move, hide, and
   show panels.  You can relocate a panel to any desired position in the stack.

Panel routines are a functional layer added to *curses*, make only high-level *curses* calls, and work anywhere *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 **FALSE** if the panel *pan* is in the panel stack, and **TRUE** if it is not.  If the

panel is a null pointer, it returns **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 if, for example, 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 "PORTABILITY" below.

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

## RETURN VALUE

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 either is null, an error is returned.

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

**NOTES**

The header file *panel.h* itself includes the header file *curses.h*.

**PORTABILITY**

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 never changed. 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 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 System V panel libraries.

**HISTORY**

A panel facility was documented in SVr4.2's *Character User Interface Programming* document.

It is not part of X/Open Curses.

A few implementations exist:

⊕     Systems based on SVr4 source code, such as 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 is a public domain implementation by Warren Tucker published in *u386mon* 2.20 (1990).

      According to Tucker, the System V 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, based on the System V documentation.

**AUTHORS**

Warren Tucker <wht@n4hgf.mt-park.ga.us> originally wrote this implementation, primarily to assist in porting *u386mon* to systems without a native panel library.

Zeyd ben-Halim repackaged it for *ncurses*.

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

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