NAME

newpad, **subpad**, **prefresh**, **pnoutrefresh**, **pechochar**, **pecho_wchar** - create and display **curses** pads

SYNOPSIS

```
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);
WINDOW *subpad(WINDOW *orig, int nlines, int ncols,
    int begin_y, int begin_x);
int prefresh(WINDOW *pad, int pminrow, int pmincol,
    int sminrow, int smincol, int smaxrow, int smaxcol);
int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol,
    int sminrow, int smincol, int smaxrow, int smaxcol);
int pechochar(WINDOW *pad, chtype ch);
int pecho_wchar(WINDOW *pad, const cchar_t *wch);
```

DESCRIPTION

newpad

The **newpad** routine creates and returns a pointer to a new pad data structure with the given number of lines, *nlines*, and columns, *ncols*. A pad is like a window, except that it is not restricted by the screen size, and is not necessarily associated with a particular part of the screen. Pads can be used when a large window is needed, and only a part of the window will be on the screen at one time. Automatic refreshes of pads (e.g., from scrolling or echoing of input) do not occur.

It is not legal to call **wrefresh** with a *pad* as an argument; the routines **prefresh** or **pnoutrefresh** should be called instead. Note that these routines require additional parameters to specify the part of the pad to be displayed and the location on the screen to be used for the display.

subpad

The **subpad** routine creates and returns a pointer to a subwindow within a pad with the given number of lines, *nlines*, and columns, *ncols*. Unlike **subwin**, which uses screen coordinates, the window is at position (*begin_x*, *begin_y*) on the pad. The window is made in the middle of the window *orig*, so that changes made to one window affect both windows. During the use of this routine, it will often be necessary to call **touchwin** or **touchline** on *orig* before calling **prefresh**.

prefresh, pnoutrefresh

The **prefresh** and **pnoutrefresh** routines are analogous to **wrefresh** and **wnoutrefresh** except that they relate to pads instead of windows. The additional parameters are needed to indicate what part of the pad and screen are involved.

- The *pminrow* and *pmincol* parameters specify the upper left-hand corner of the rectangle to be displayed in the pad.

- The *sminrow*, *smincol*, *smaxrow*, and *smaxcol* parameters specify the edges of the rectangle to be displayed on the screen.

The lower right-hand corner of the rectangle to be displayed in the pad is calculated from the screen coordinates, since the rectangles must be the same size. Both rectangles must be entirely contained within their respective structures. Negative values of *pminrow*, *pmincol*, *sminrow*, or *smincol* are treated as if they were zero.

pechochar

The **pechochar** routine is functionally equivalent to a call to **addch** followed by a call to **refresh**(3X), a call to **waddch** followed by a call to **wrefresh**, or a call to **waddch** followed by a call to **prefresh**. The knowledge that only a single character is being output is taken into consideration and, for non-control characters, a considerable performance gain might be seen by using these routines instead of their equivalents. In the case of **pechochar**, the last location of the pad on the screen is reused for the arguments to **prefresh**.

pecho_wchar

The **pecho_wchar** function is the analogous wide-character form of **pechochar**. It outputs one character to a pad and immediately refreshes the pad. It does this by a call to **wadd_wch** followed by a call to **prefresh**.

RETURN VALUE

Routines that return an integer return **ERR** upon failure and **OK** (SVr4 only specifies "an integer value other than **ERR**") upon successful completion.

Routines that return pointers return **NULL** on error, and set **errno** to **ENOMEM**.

X/Open does not define any error conditions. In this implementation

prefresh and **pnoutrefresh**

return an error if the window pointer is null, or if the window is not really a pad or if the area to refresh extends off-screen or if the minimum coordinates are greater than the maximum.

pechochar

returns an error if the window is not really a pad, and the associated call to **wechochar** returns an error.

pecho_wchar
returns an error if the window is not really a pad, and the associated call to **wecho_wchar** returns an error.

NOTES

Note that **pechochar** may be a macro.

PORTABILITY

BSD curses has no *pad* feature.

SVr2 curses (1986) provided the **newpad** and related functions, documenting them in a single line each. SVr3 (1987) provided more extensive documentation.

The documentation does not explain the term *pad*. However, the Apollo *Aegis* workstation operating system supported a graphical *pad* feature:

- These graphical pads could be much larger than the computer's display.
- The read-only output from a command could be scrolled back to inspect, and select text from the pad.

The two uses may be related.

The XSI Curses standard, Issue 4 describes these functions, without significant change from the SVr3 documentation. It describes no error conditions. The behavior of **subpad** if the parent window is not a pad is undocumented, and is not checked by the vendor Unix implementations:

- SVr4 curses sets a flag in the **WINDOW** structure in **newpad** which tells if the window is a *pad*.

  However, it uses this information only in **waddch** (to decide if it should call **wrefresh**) and **wscrl** (to avoid scrolling a pad), and does not check in **wrefresh** to ensure that the pad is refreshed properly.

- Solaris X/Open Curses checks if a window is a pad in **wnoutrefresh**, returning **ERR** in that case.

  However, it only sets the flag for subwindows if the parent window is a pad. Its **newpad** function does not set this information. Consequently, the check will never fail.

  It makes no comparable check in **pnoutrefresh**, though interestingly enough, a comment in the source code states that the lack of a check was an MKS extension.

- NetBSD 7 curses sets a flag in the **WINDOW** structure for **newpad** and **subpad**, using this to help with the distinction between **wnoutrefresh** and **pnoutrefresh**.

  It does not check for the case where a subwindow is created in a pad using **subwin** or **derwin**.

  The **dupwin** function returns a regular window when duplicating a pad. Likewise, **getwin** always returns a window, even if the saved data was from a pad.

This implementation

- sets a flag in the **WINDOW** structure for **newpad** and **subpad**,
- allows a **subwin** or **derwin** call to succeed having a pad parent by forcing the subwindow to be a pad,
- checks in both **wnoutrefresh** and **pnoutrefresh** to ensure that pads and windows are handled distinctly, and
- ensures that **dupwin** and **getwin** treat pads versus windows consistently.

SEE ALSO

curses(3X), **curs_refresh**(3X), **curs_touch**(3X), **curs_addch**(3X).