NAME

doupdate, **redrawwin**, **refresh**, **wnoutrefresh**, **wredrawln**, **wrefresh** - refresh *curses* windows or lines thereupon

SYNOPSIS

```
#include <curses.h>

int refresh(void);
int wrefresh(WINDOW *win);
int wnoutrefresh(WINDOW *win);
int doupdate(void);

int redrawwin(WINDOW *win);
int wredrawln(WINDOW *win, int beg_line, int num_lines);
```

DESCRIPTION

refresh, wrefresh

The **refresh** and **wrefresh** routines (or **wnoutrefresh** and **doupdate**) must be called to get actual output to the terminal, as other routines merely manipulate data structures. The routine **wrefresh** copies the named window to the *physical screen*, taking into account what is already there to do optimizations. The **refresh** routine is the same, using **stdscr** as the default window. Unless **leaveok**(3X) has been enabled, the physical cursor of the terminal is left at the location of the cursor for that window.

wnoutrefresh, doupdate

The **wnoutrefresh** and **doupdate** routines allow multiple updates with more efficiency than **wrefresh** alone. In addition to all the window structures, **curses** keeps two data structures representing the terminal screen:

- a *physical screen*, describing what is actually on the screen, and
- a *virtual screen*, describing what the programmer wants to have on the screen.

The routine **wrefresh** works by

- first calling **wnoutrefresh**, which copies the named window to the *virtual screen*, and
- then calling **doupdate**, which compares the *virtual screen* to the *physical screen* and does the actual update.

If the programmer wishes to output several windows at once, a series of calls to **wrefresh** results in

alternating calls to **wnoutrefresh** and **doupdate**, causing several bursts of output to the screen. By first calling **wnoutrefresh** for each window, it is then possible to call **doupdate** once, resulting in only one burst of output, with fewer total characters transmitted and less CPU time used.

If the *win* argument to **wrefresh** is the *physical screen* (i.e., the global variable **curscr**), the screen is immediately cleared and repainted from scratch.

The phrase "copies the named window to the virtual screen" above is ambiguous. What actually happens is that all *touched* (changed) lines in the window are copied to the virtual screen. This affects programs that use overlapping windows; it means that if two windows overlap, you can refresh them in either order and the overlap region will be modified only when it is explicitly changed. (But see the section on **PORTABILITY** below for a warning about exploiting this behavior.)

wredrawln, redrawwin

The **wredrawln** routine indicates to **curses** that some screen lines are corrupted and should be thrown away before anything is written over them. It touches the indicated lines (marking them changed). The routine **redrawwin** touches the entire window.

RETURN VALUE

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

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

wnoutrefresh

returns an error if the window pointer is null, or if the window is really a pad.

wredrawln

returns an error if the associated call to **touchln** returns an error.

NOTES

Note that **refresh** and **redrawwin** may be macros.

PORTABILITY

X/Open Curses, Issue 4 describes these functions.

Whether **wnoutrefresh** copies to the virtual screen the entire contents of a window or just its changed portions has never been well-documented in historic curses versions (including SVr4). It might be unwise to rely on either behavior in programs that might have to be linked with other curses implementations. Instead, you can do an explicit **touchwin** before the **wnoutrefresh** call to guarantee an

entire-contents copy anywhere.

SEE ALSO

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