

NAME

alloc_pair, **find_pair**, **free_pair** - new curses color-pair functions

SYNOPSIS

```
#include <curses.h>
```

```
int alloc_pair(int fg, int bg);
```

```
int find_pair(int fg, int bg);
```

```
int free_pair(int pair);
```

DESCRIPTION

These functions are an extension to the curses library. They permit an application to dynamically allocate a color pair using the foreground/background colors rather than assign a fixed color pair number, and return an unused pair to the pool.

The number of colors may be related to the number of possible color pairs for a given terminal, or it may not:

- ⊕ While almost all terminals allow setting the color *attributes* independently, it is unlikely that your terminal allows you to modify the attributes of a given character cell without rewriting it. That is, the foreground and background colors are applied as a pair.
- ⊕ Color pairs are the curses library's way of managing a color palette on a terminal. If the library does not keep track of the *combinations* of colors which are displayed, it will be inefficient.
- ⊕ For simple terminal emulators with only a few dozen color combinations, it is convenient to use the maximum number of combinations as the limit on color pairs:

COLORS * COLORS

- ⊕ Terminals which support *default colors* distinct from "ANSI colors" add to the possible combinations, producing this total:

$$(\text{COLORS} + 1) * (\text{COLORS} + 1)$$

- ⊕ An application might use up to a few dozen color pairs to implement a predefined color scheme.

Beyond that lies in the realm of programs using the foreground and background colors for "ASCII art" (or some other non-textual application).

Also beyond those few dozen pairs, the required size for a table to represent the combinations grows rapidly with an increasing number of colors.

These functions allow a developer to let the screen library manage color pairs.

alloc_pair

The **alloc_pair** function accepts parameters for foreground and background color, and checks if that color combination is already associated with a color pair.

- ⊕ If the combination already exists, **alloc_pair** returns the existing pair.
- ⊕ If the combination does not exist, **alloc_pair** allocates a new color pair and returns that.
- ⊕ If the table fills up, **alloc_pair** discards the least-recently allocated entry using **free_pair** and allocates a new color pair.

All of the color pairs are allocated from a table of possible color pairs. The size of the table is determined by the terminfo *pairs* capability. The table is shared with **init_pair**; in fact **alloc_pair** calls **init_pair** after updating the ncurses library's fast index to the colors versus color pairs.

find_pair

The **find_pair** function accepts parameters for foreground and background color, and checks if that color combination is already associated with a color pair, returning the pair number if it has been allocated. Otherwise it returns -1.

free_pair

Marks the given color pair as unused, i.e., like color pair 0.

RETURN VALUE

The **alloc_pair** function returns a color pair number in the range 1 through **COLOR_PAIRS**-1, unless it encounters an error updating its fast index to the color pair values, preventing it from allocating a color pair. In that case, it returns -1.

The **find_pair** function returns a color pair number if the given color combination has been associated with a color pair, or -1 if not.

Likewise, **free_pair** returns **OK** unless it encounters an error updating the fast index or if no such color pair is in use.

PORTABILITY

new_pair(3X)

new_pair(3X)

These routines are specific to ncurses. They were not supported on Version 7, BSD or System V implementations. It is recommended that any code depending on them be conditioned using NCURSES_VERSION.

SEE ALSO

curs_color(3X).

AUTHOR

Thomas Dickey.

new_pair(3X)