

**NAME**

**slk\_init**, **slk\_set**, **slk\_wset**, **slk\_refresh**, **slk\_noutrefresh**, **slk\_label**, **slk\_clear**, **slk\_restore**, **slk\_touch**, **slk\_atron**, **slk\_attrset**, **slk\_attroff**, **slk\_attr\_on**, **slk\_attr\_set**, **slk\_attr\_off**, **slk\_attr**, **slk\_color**, **extended\_slk\_color** - curses soft label routines

**SYNOPSIS**

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

```
int slk_init(int fmt);
```

```
int slk_set(int labnum, const char *label, int fmt);
```

```
/* extension */
```

```
int slk_wset(int labnum, const wchar_t *label, int fmt);
```

```
char *slk_label(int labnum);
```

```
int slk_refresh(void);
```

```
int slk_noutrefresh(void);
```

```
int slk_clear(void);
```

```
int slk_restore(void);
```

```
int slk_touch(void);
```

```
int slk_atron(const chtype attrs);
```

```
int slk_attroff(const chtype attrs);
```

```
int slk_attrset(const chtype attrs);
```

```
int slk_attr_on(attr_t attrs, void* opts);
```

```
int slk_attr_off(const attr_t attrs, void * opts);
```

```
int slk_attr_set(const attr_t attrs, short pair, void* opts);
```

```
attr_t slk_attr(void);
```

```
int slk_color(short pair);
```

```
/* extension */
```

```
int extended_slk_color(int pair);
```

**DESCRIPTION**

The **slk\*** functions manipulate the set of soft function-key labels that exist on many terminals. For those terminals that do not have soft labels, **curses** takes over the bottom line of **stdscr**, reducing the size of **stdscr** and the variable **LINES**. **curses** standardizes on eight labels of up to eight characters each. In addition to this, the ncurses implementation supports a mode where it simulates 12 labels of

up to five characters each. This is useful for PC-like enduser devices. `ncurses` simulates this mode by taking over up to two lines at the bottom of the screen; it does not try to use any hardware support for this mode.

### Initialization

The `slk_init` routine must be called before `initscr` or `newterm` is called. If `initscr` eventually uses a line from `stdscr` to emulate the soft labels, then `fmt` determines how the labels are arranged on the screen:

- 0** indicates a 3-2-3 arrangement of the labels.
- 1** indicates a 4-4 arrangement
- 2** indicates the PC-like 4-4-4 mode.
- 3** is again the PC-like 4-4-4 mode, but in addition an index line is generated, helping the user to identify the key numbers easily.

### Labels

The `slk_set` routine (and the `slk_wset` routine for the wide-character library) has three parameters:

*labnum*

is the label number, from **1** to **8** (12 for `fmt` in `slk_init` is **2** or **3**);

*label* is be the string to put on the label, up to eight (five for `fmt` in `slk_init` is **2** or **3**) characters in length. A null string or a null pointer sets up a blank label.

*fmt* is either **0**, **1**, or **2**, indicating whether the label is to be left-justified, centered, or right-justified, respectively, within the label.

The `slk_label` routine returns the current label for label number *labnum*, with leading and trailing blanks stripped.

### Screen updates

The `slk_refresh` and `slk_noutrefresh` routines correspond to the `wrefresh` and `wnoutrefresh` routines.

The `slk_clear` routine clears the soft labels from the screen.

The `slk_restore` routine restores the soft labels to the screen after a `slk_clear` has been performed.

The `slk_touch` routine forces all the soft labels to be output the next time a `slk_noutrefresh` is

performed.

### Video attributes

The **slk\_attron**, **slk\_attrset**, **slk\_attroff** and **slk\_attr** routines correspond to **attron**, **attrset**, **attroff** and **attr\_get**, respectively. They have an effect only if soft labels are simulated on the bottom line of the screen. The default highlight for soft keys is A\_STANDOUT (as in System V curses, which does not document this fact).

### Colors

The **slk\_color** routine corresponds to **color\_set**. It has an effect only if soft labels are simulated on the bottom line of the screen.

Because **slk\_color** accepts only **short** (signed 16-bit integer) values, this implementation provides **extended\_slk\_color** which accepts an integer value, e.g., 32-bits.

### RETURN VALUE

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

X/Open defines no error conditions. In this implementation

#### **slk\_attr**

returns the attribute used for the soft keys.

#### **slk\_attroff**, **slk\_attron**, **slk\_clear**, **slk\_noutrefresh**, **slk\_refresh**, **slk\_touch**

return an error if the terminal or the softkeys were not initialized.

#### **slk\_attrset**

returns an error if the terminal or the softkeys were not initialized.

#### **slk\_attr\_set**

returns an error if the terminal or the softkeys were not initialized, or the color pair is outside the range 0..COLOR\_PAIRS-1.

#### **slk\_color**

returns an error if the terminal or the softkeys were not initialized, or the color pair is outside the range 0..COLOR\_PAIRS-1.

#### **slk\_init**

returns an error if the format parameter is outside the range 0..3.

**slk\_label**

returns **NULL** on error.

**slk\_set**

returns an error if the terminal or the softkeys were not initialized, or the *labnum* parameter is outside the range of label counts, or if the format parameter is outside the range 0..2, or if memory for the labels cannot be allocated.

**HISTORY**

SVr3 introduced these functions:

- slk\_clear
- slk\_init
- slk\_label
- slk\_noutrefresh
- slk\_refresh
- slk\_restore
- slk\_set
- slk\_touch

SVr4 added these functions:

- slk\_attroff
- slk\_attron
- slk\_attrset
- slk\_start

X/Open Curses added these:

- slk\_attr\_off
- slk\_attr\_on
- slk\_attr\_set
- slk\_color
- slk\_wset

**EXTENSIONS**

X/Open Curses documents the *opts* argument as reserved for future use, saying that it must be null. This implementation uses that parameter in ABI 6 for the functions which have a color-pair parameter to support extended color pairs.

For functions which modify the color, e.g., **slk\_attr\_set**, if *opts* is set it is treated as a pointer to **int**, and used to set the color pair instead of the **short** pair parameter.

## NOTES

Most applications would use **slk\_noutrefresh** because a **wrefresh** is likely to follow soon.

## PORTABILITY

The XSI Curses standard, Issue 4, described the soft-key functions, with some differences from SVr4 curses:

- ⊕ It added functions like the SVr4 attribute-manipulation functions **slk\_attron**, **slk\_attroff**, **slk\_attrset**, but which use **attr\_t** parameters (rather than **chtype**), along with a reserved *opts* parameter.

Two of these new functions (unlike the SVr4 functions) have no provision for color: **slk\_attr\_on** and **slk\_attr\_off**.

The third function (**slk\_attr\_set**) has a color-pair parameter.

- ⊕ It added **const** qualifiers to parameters (unnecessarily), and
- ⊕ It added **slk\_color**.

The format codes **2** and **3** for **slk\_init** and the function **slk\_attr** are specific to ncurses.

X/Open Curses does not specify a limit for the number of colors and color pairs which a terminal can support. However, in its use of **short** for the parameters, it carries over SVr4's implementation detail for the compiled terminfo database, which uses signed 16-bit numbers. This implementation provides extended versions of those functions which use **short** parameters, allowing applications to use larger color- and pair-numbers.

## SEE ALSO

**curses(3X)**, **curs\_attr(3X)**, **curs\_initscr(3X)**, **curs\_refresh(3X)**, **curs\_variables(3X)**.