curs getstr(3X) curs getstr(3X)

# **NAME**

**getstr**, **getnstr**, **wgetstr**, **wgetnstr**, **mvgetstr**, **mvwgetstr**, **mvwgetnstr** - accept character strings from **curses** terminal keyboard

# **SYNOPSIS**

#include <curses.h>

```
int getstr(char *str);
int getnstr(char *str, int n);
int wgetstr(WINDOW *win, char *str);
int wgetnstr(WINDOW *win, char *str, int n);
int mvgetstr(int y, int x, char *str);
int mvgetstr(WINDOW *win, int y, int x, char *str);
int mvgetnstr(int y, int x, char *str, int n);
int mvwgetnstr(WINDOW *win, int y, int x, char *str, int n);
```

## DESCRIPTION

The function **getstr** is equivalent to a series of calls to **getch**, until a newline or carriage return is received (the terminating character is not included in the returned string). The resulting value is placed in the area pointed to by the character pointer *str*, followed by a NUL.

wgetnstr reads at most *n* characters, thus preventing a possible overflow of the input buffer. Any attempt to enter more characters (other than the terminating newline or carriage return) causes a beep. Function keys also cause a beep and are ignored. The getnstr function reads from the *stdscr* default window.

The user's erase and kill characters are interpreted. If keypad mode is on for the window, **KEY\_LEFT** and **KEY\_BACKSPACE** are both considered equivalent to the user's kill character.

Characters input are echoed only if **echo** is currently on. In that case, backspace is echoed as deletion of the previous character (typically a left motion).

# **RETURN VALUE**

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

X/Open defines no error conditions.

In this implementation, these functions return an error if the window pointer is null, or if its timeout

curs\_getstr(3X) curs\_getstr(3X)

expires without having any data.

This implementation provides an extension as well. If a **SIGWINCH** interrupts the function, it will return **KEY\_RESIZE** rather than **OK** or **ERR**.

Functions with a "mv" prefix first perform a cursor movement using **wmove**, and return an error if the position is outside the window, or if the window pointer is null.

# **NOTES**

Note that **getstr**, **mvgetstr**, and **mvwgetstr** may be macros.

# **PORTABILITY**

These functions are described in the XSI Curses standard, Issue 4. They read single-byte characters only. The standard does not define any error conditions. This implementation returns **ERR** if the window pointer is null, or if the lower-level **wgetch**(3X) call returns an **ERR**.

SVr3 and early SVr4 curses implementations did not reject function keys; the SVr4.0 documentation claimed that "special keys" (such as function keys, "home" key, "clear" key, *etc.*) are "interpreted", without giving details. It lied. In fact, the "character" value appended to the string by those implementations was predictable but not useful (being, in fact, the low-order eight bits of the key's KEY\_value).

The functions **getnstr**, **mvgetnstr**, and **mvwgetnstr** were present but not documented in SVr4.

X/Open Curses, Issue 5 (2007) stated that these functions "read at most *n* bytes" but did not state whether the terminating NUL is counted in that limit. X/Open Curses, Issue 7 (2009) changed that to say they "read at most *n*-1 bytes" to allow for the terminating NUL. As of 2018, some implementations do, some do not count it:

- Φ ncurses 6.1 and PDCurses do not count the NUL in the given limit, while
- Solaris SVr4 and NetBSD curses count the NUL as part of the limit.
- Solaris xcurses provides both: its wide-character **wget\_nstr** reserves a NUL, but its **wgetnstr** does not count the NUL consistently.

In SVr4 curses, a negative value of n tells **wgetnstr** to assume that the caller's buffer is large enough to hold the result, i.e., to act like **wgetstr**. X/Open Curses does not mention this (or anything related to negative or zero values of n), however most implementations use the feature, with different limits:

curs\_getstr(3X) curs\_getstr(3X)

Φ Solaris SVr4 curses and PDCurses limit the result to 255 bytes. Other Unix systems than Solaris are likely to use the same limit.

- Solaris xcurses limits the result to **LINE\_MAX** bytes.
- NetBSD 7 assumes no particular limit for the result from **wgetstr**. However, it limits the **wgetnstr** parameter *n* to ensure that it is greater than zero.

A comment in NetBSD's source code states that this is specified in SUSv2.

- $\oplus$  ncurses (before 6.2) assumes no particular limit for the result from **wgetstr**, and treats the *n* parameter of **wgetnstr** like SVr4 curses.
- ncurses 6.2 uses **LINE\_MAX**, or a larger (system-dependent) value which the **sysconf** function may provide. If neither **LINE\_MAX** or **sysconf** is available, ncurses uses the POSIX value for **LINE\_MAX** (a 2048 byte limit). In either case, it reserves a byte for the terminating NUL.

## SEE ALSO

curses(3X),  $curs\_getch(3X)$ ,  $curs\_variables(3X)$ .