NAME

scanw, **wscanw**, **mvscanw**, **mvwscanw**, **vwscanw**, **vw_scanw** - convert formatted input from a **curses** window

SYNOPSIS

```
#include <curses.h>

int scanw(const char *fmt, ...);
int wscanw(WINDOW *win, const char *fmt, ...);
int mvscanw(int y, int x, const char *fmt, ...);
int mvwscanw(WINDOW *win, int y, int x, const char *fmt, ...);

int vw_scanw(WINDOW *win, const char *fmt, va_list varglist);

/* obsolete */
int vwscanw(WINDOW *win, const char *fmt, va_list varglist);
```

DESCRIPTION

The **scanw**, **wscanw** and **mvscanw** routines are analogous to **scanf** [see **scanf**(3)]. The effect of these routines is as though **wgetstr** were called on the window, and the resulting line used as input for **sscanf**(3). Fields which do not map to a variable in the *fmt* field are lost.

The **vwscanw** and **vw_scanw** routines are analogous to **vscanf**(3). They perform a **wscanw** using a variable argument list. The third argument is a *va_list*, a pointer to a list of arguments, as defined in **<stdarg.h>**.

RETURN VALUE

vwscanw returns **ERR** on failure and an integer equal to the number of fields scanned on success.

Applications may use the return value from the **scanw**, **wscanw**, **mvscanw** and **mvwscanw** routines to determine the number of fields which were mapped in the call.

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.

HISTORY

While **scanw** was implemented in 4BSD, none of the BSD releases used it until 4.4BSD (in a game). That early version of curses was before the ANSI C standard. It did not use <varargs.h>, though that was available. In 1991 (a couple of years after SVr4 was generally available, and after the C standard was published), other developers updated the library, using <stdarg.h> internally in 4.4BSD curses.

Even with this improvement, BSD curses did not use function prototypes (or even declare functions) in the <curses.h> header until 1992.

SVr2 documented **scanw**, **wscanw** tersely as "scanf through *stdscr*" and tersely as "scanf through *win*", respectively.

SVr3 added **mvscanw**, and **mvwscanw**, with a three-line summary saying that they were analogous to **scanf**(3), explaining that the string which would be output from **scanf**(3) would instead be output using **waddstr** on the given window. SVr3 also added **vwscanw**, saying that the third parameter is a **va_list**, defined in <varargs.h>, and referring the reader to the manual pages for *varargs* and *vprintf* for detailed descriptions. (Because the SVr3 documentation does not mention *vscanf*, that reference to *vprintf* may not be an error).

SVr4 added no new variations of **scanw**, but provided for using <varargs.h> or <stdarg.h> to define the **va_list** type.

X/Open Curses added **vw_scanw** to replace **vwscanw**, stating that its **va_list** definition requires <stdarg.h>.

PORTABILITY

In this implementation, **vw_scanw** and **vwscanw** are equivalent, to support legacy applications. However, the latter (**vwscanw**) is obsolete:

- The XSI Curses standard, Issue 4 described these functions, noting that the function **vwscanw** is marked TO BE WITHDRAWN, and is to be replaced by a function **vw_scanw** using the **<stdarg.h>** interface.

- The Single Unix Specification, Version 2 states that **vw_scanw** is preferred to **vwscanw** since the latter requires including **<varargs.h>**, which cannot be used in the same file as **<stdarg.h>**. This implementation uses **<stdarg.h>** for both, because that header is included in **<curses.h**>.

- X/Open Curses, Issue 5 (December 2007) marked **vwscanw** (along with **vwprintw** and the termcap interface) as withdrawn.

Both XSI and The Single Unix Specification, Version 2 state that these functions return **ERR** or **OK**.

- Since the underlying **scanf**(3) can return the number of items scanned, and the SVr4 code was documented to use this feature, this is probably an editing error which was introduced in XSI, rather than being done intentionally.

- This implementation returns the number of items scanned, for compatibility with SVr4 curses. As of 2018, NetBSD curses also returns the number of items scanned. Both ncurses and NetBSD curses call **vsscanf** to scan the string, which returns **EOF** on error.

- Portable applications should only test if the return value is **ERR**, since the **OK** value (zero) is likely to be misleading.

  One possible way to get useful results would be to use a "%n" conversion at the end of the format string to ensure that something was processed.

SEE ALSO

curses(3X), **curs_getstr**(3X), **curs_printw**(3X), **curs_termcap**(3X), **scanf**(3).