**NAME**

curs_getch, getch, wgetch, mvgetch, mvwgetch, ungetch – get (or push back) characters from curses terminal keyboard

**SYNOPSIS**

**cc** [ *flag* ... ] *file* ... **–lcurses** [ *library* .. ]

**#include <curses.h>**

**int getch(void);**

**int wgetch(WINDOW** **win***);**

**int mvgetch(int** *y***, int** *x***);**

**int mvwgetch(WINDOW** **win***, int** *y***, int** *x***);**

**int ungetch(int** *ch***);**

**MT-LEVEL**

Unsafe

**DESCRIPTION**

With the **getch( )**, **wgetch( )**, **mvgetch( )**, and **mvwgetch( )** routines a character is read from the terminal associated with the window. In no-delay mode, if no input is waiting, the value **ERR** is returned. In delay mode, the program waits until the system passes text through to the program. Depending on the setting of **cbreak( )**, this is after one character (cbreak mode), or after the first newline (nocbreak mode). In half-delay mode, the program waits until a character is typed or the specified timeout has been reached. Unless **noecho( )** has been set, the character will also be echoed into the designated window.

If the window is not a pad, and it has been moved or modified since the last call to **wrefresh( )**, **wrefresh( )** will be called before another character is read.

If **keypad( )** is **TRUE**, and a function key is pressed, the token for that function key is returned instead of the raw characters. Possible function keys are defined in **<curses.h>** with integers beginning with **0401**, whose names begin with **KEY_**. If a character that could be the beginning of a function key (such as escape) is received, **curses** sets a timer. If the remainder of the sequence does not come in within the designated time, the character is passed through; otherwise, the function key value is returned. For this reason, many terminals experience a delay between the time a user presses the escape key and the escape is returned to the program. Since tokens returned by these routines are outside the ASCII range, they are not printable.

The **ungetch( )** routine places *ch* back onto the input queue to be returned by the next call to **wgetch( )**.

**Function Keys**

The following function keys, defined in **<curses.h>**, might be returned by **getch( )** if **keypad( )** has been enabled. Note that not all of these may be supported on a particular terminal if the terminal does not transmit a unique code when the key is pressed or if the definition for the key is not present in the *terminfo* database.

| *Name* | *Key name* |
|---|---|
| KEY_BREAK | Break key |
| KEY_DOWN | The four arrow keys . . . |
| KEY_UP | |
| KEY_LEFT | |
| KEY_RIGHT | |
| KEY_HOME | Home key (upward+left arrow) |
| KEY_BACKSPACE | Backspace |
| KEY_F0 | Function keys; space for 64 keys is reserved. |
| KEY_F(*n*) | For $0 \le n \le 63$ |
| KEY_DL | Delete line |
| KEY_IL | Insert line |
| KEY_DC | Delete character |
| KEY_IC | Insert char or enter insert mode |
| KEY_EIC | Exit insert char mode |
| KEY_CLEAR | Clear screen |
| KEY_EOS | Clear to end of screen |
| KEY_EOL | Clear to end of line |
| KEY_SF | Scroll 1 line forward |
| KEY_SR | Scroll 1 line backward (reverse) |
| KEY_NPAGE | Next page |
| KEY_PPAGE | Previous page |
| KEY_STAB | Set tab |
| KEY_CTAB | Clear tab |
| KEY_CATAB | Clear all tabs |
| KEY_ENTER | Enter or send |
| KEY_SRESET | Soft (partial) reset |
| KEY_RESET | Reset or hard reset |
| KEY_PRINT | Print or copy |
| KEY_LL | Home down or bottom (lower left). Keypad is arranged like this:<br>**A1 up A3**<br>**left B2 right**<br>**C1 down C3** |
| KEY_A1 | Upper left of keypad |
| KEY_A3 | Upper right of keypad |
| KEY_B2 | Center of keypad |
| KEY_C1 | Lower left of keypad |
| KEY_C3 | Lower right of keypad |
| KEY_BTAB | Back tab key |
| KEY_BEG | Beg(inning) key |
| KEY_CANCEL | Cancel key |
| KEY_CLOSE | Close key |
| KEY_COMMAND | Cmd (command) key |
| KEY_COPY | Copy key |
| KEY_CREATE | Create key |
| KEY_END | End key |
| KEY_EXIT | Exit key |
| KEY_FIND | Find key |

| | |
|---|---|
| KEY_HELP | Help key |
| KEY_MARK | Mark key |
| KEY_MESSAGE | Message key |
| KEY_MOVE | Move key |
| KEY_NEXT | Next object key |
| KEY_OPEN | Open key |
| KEY_OPTIONS | Options key |
| KEY_PREVIOUS | Previous object key |
| KEY_REDO | Redo key |
| KEY_REFERENCE | Reference key |
| KEY_REFRESH | Refresh key |
| KEY_REPLACE | Replace key |
| KEY_RESTART | Restart key |
| KEY_RESUME | Resume key |
| KEY_SAVE | Save key |
| KEY_SBEG | Shifted beginning key |
| KEY_SCANCEL | Shifted cancel key |
| KEY_SCOMMAND | Shifted command key |
| KEY_SCOPY | Shifted copy key |
| KEY_SCREATE | Shifted create key |
| KEY_SDC | Shifted delete char key |
| KEY_SDL | Shifted delete line key |
| KEY_SELECT | Select key |
| KEY_SEND | Shifted end key |
| KEY_SEOL | Shifted clear line key |
| KEY_SEXIT | Shifted exit key |
| KEY_SFIND | Shifted find key |
| KEY_SHELP | Shifted help key |
| KEY_SHOME | Shifted home key |
| KEY_SIC | Shifted input key |
| KEY_SLEFT | Shifted left arrow key |
| KEY_SMESSAGE | Shifted message key |
| KEY_SMOVE | Shifted move key |
| KEY_SNEXT | Shifted next key |
| KEY_SOPTIONS | Shifted options key |
| KEY_SPREVIOUS | Shifted prev key |
| KEY_SPRINT | Shifted print key |
| KEY_SREDO | Shifted redo key |
| KEY_SREPLACE | Shifted replace key |
| KEY_SRIGHT | Shifted right arrow |
| KEY_SRSUME | Shifted resume key |
| KEY_SSAVE | Shifted save key |
| KEY_SSUSPEND | Shifted suspend key |
| KEY_SUNDO | Shifted undo key |
| KEY_SUSPEND | Suspend key |
| KEY_UNDO | Undo key |

**RETURN VALUES**

All routines return the integer **ERR** upon failure. The **ungetch( )** routine returns an integer value other than **ERR** upon successful completion. The other routines return the next input character or function key code upon successful completion.

**SEE ALSO**

**curs_inopts**(3X), **curs_move**(3X), **curs_refresh**(3X), **curses**(3X)

**NOTES**

The header **<curses.h>** automatically includes the headers **<stdio.h>** and **<unctrl.h>**.

Use of the escape key for a single character function is discouraged.

When using **getch( )**, **wgetch( )**, **mvgetch( )**, or **mvwgetch( )**, **nocbreak** mode (**nocbreak( )**) and **echo** mode (**echo( )**) should not be used at the same time. Depending on the state of the tty driver when each character is typed, the program may produce undesirable results.

Note that **getch( )**, **mvgetch( )**, and **mvwgetch( )** may be macros.