

**NAME**

**menu\_driver** - command-processing loop of the menu system

**SYNOPSIS**

```
#include <menu.h>
```

```
int menu_driver(MENU *menu, int c);
```

**DESCRIPTION**

Once a menu has been posted (displayed), you should funnel input events to it through **menu\_driver**.

This routine has three major input cases:

- ⊕ The input is a form navigation request. Navigation request codes are constants defined in **<form.h>**, which are distinct from the key- and character codes returned by **wgetch(3X)**.
- ⊕ The input is a printable character. Printable characters (which must be positive, less than 256) are checked according to the program's locale settings.
- ⊕ The input is the **KEY\_MOUSE** special key associated with an mouse event.

The menu driver requests are as follows:

**REQ\_LEFT\_ITEM**

Move left to an item.

**REQ\_RIGHT\_ITEM**

Move right to an item.

**REQ\_UP\_ITEM**

Move up to an item.

**REQ\_DOWN\_ITEM**

Move down to an item.

**REQ\_SCR\_ULINE**

Scroll up a line.

**REQ\_SCR\_DLINE**

Scroll down a line.

REQ\_SCR\_DPAGE

Scroll down a page.

REQ\_SCR\_UPAGE

Scroll up a page.

REQ\_FIRST\_ITEM

Move to the first item.

REQ\_LAST\_ITEM

Move to the last item.

REQ\_NEXT\_ITEM

Move to the next item.

REQ\_PREV\_ITEM

Move to the previous item.

REQ\_TOGGLE\_ITEM

Select/deselect an item.

REQ\_CLEAR\_PATTERN

Clear the menu pattern buffer.

REQ\_BACK\_PATTERN

Delete the previous character from the pattern buffer.

REQ\_NEXT\_MATCH

Move to the next item matching the pattern match.

REQ\_PREV\_MATCH

Move to the previous item matching the pattern match.

If the second argument is a printable character, the code appends it to the pattern buffer and attempts to move to the next item matching the new pattern. If there is no such match, **menu\_driver** returns **E\_NO\_MATCH** and deletes the appended character from the buffer.

If the second argument is one of the above pre-defined requests, the corresponding action is performed.

## Mouse Handling

If the second argument is the `KEY_MOUSE` special key, the associated mouse event is translated into one of the above pre-defined requests. Currently only clicks in the user window (e.g., inside the menu display area or the decoration window) are handled.

If you click above the display region of the menu:

- ⊕ a `REQ_SCR_ULINE` is generated for a single click,
- ⊕ a `REQ_SCR_UPAGE` is generated for a double-click and
- ⊕ a `REQ_FIRST_ITEM` is generated for a triple-click.

If you click below the display region of the menu:

- ⊕ a `REQ_SCR_DLINE` is generated for a single click,
- ⊕ a `REQ_SCR_DPAGE` is generated for a double-click and
- ⊕ a `REQ_LAST_ITEM` is generated for a triple-click.

If you click at an item inside the display area of the menu:

- ⊕ the menu cursor is positioned to that item.
- ⊕ If you double-click an item a `REQ_TOGGLE_ITEM` is generated and **`E_UNKNOWN_COMMAND`** is returned. This return value makes sense, because a double click usually means that an item-specific action should be returned. It is exactly the purpose of this return value to signal that an application specific command should be executed.
- ⊕ If a translation into a request was done, **`menu_driver`** returns the result of this request.

If you clicked outside the user window or the mouse event could not be translated into a menu request an **`E_REQUEST_DENIED`** is returned.

### Application-defined Commands

If the second argument is neither printable nor one of the above pre-defined menu requests or `KEY_MOUSE`, the drive assumes it is an application-specific command and returns **`E_UNKNOWN_COMMAND`**. Application-defined commands should be defined relative to **`MAX_COMMAND`**, the maximum value of these pre-defined requests.

**RETURN VALUE**

**menu\_driver** return one of the following error codes:

**E\_OK**

The routine succeeded.

**E\_SYSTEM\_ERROR**

System error occurred (see **errno(3)**).

**E\_BAD\_ARGUMENT**

Routine detected an incorrect or out-of-range argument.

**E\_BAD\_STATE**

Routine was called from an initialization or termination function.

**E\_NOT\_POSTED**

The menu has not been posted.

**E\_UNKNOWN\_COMMAND**

The menu driver code saw an unknown request code.

**E\_NO\_MATCH**

Character failed to match.

**E\_REQUEST\_DENIED**

The menu driver could not process the request.

**PORTABILITY**

These routines emulate the System V menu library. They were not supported on Version 7 or BSD versions. The support for mouse events is *ncurses* specific.

**AUTHORS**

Juergen Pfeifer. Manual pages and adaptation for new curses by Eric S. Raymond.

**SEE ALSO**

**curses(3X)**, **getch(3X)**, **menu(3X)**