NAME

form_field_buffer - field buffer control

SYNOPSIS

```
#include <form.h>

int set_field_buffer(FIELD *field, int buf, const char *value);
char *field_buffer(const FIELD *field, int buffer);

int set_field_status(FIELD *field, bool status);
bool field_status(const FIELD *field);

int set_max_field(FIELD *field, int max);
```

DESCRIPTION

The function **set_field_buffer** sets the numbered buffer of the given field to contain a given string:

- Buffer 0 is the displayed value of the field.
- Other numbered buffers may be allocated by applications through the **nbuf** argument of (see **form_field_new**(3X)) but are not manipulated by the forms library.

The function **field_buffer** returns a pointer to the contents of the given numbered buffer:

- The buffer contents always have the same length, and are padded with trailing spaces as needed to ensure this length is the same.
- The buffer may contain leading spaces, depending on how it was set.
- The buffer contents are set with **set_field_buffer**, or as a side effect of any editing operations on the corresponding field.
- Editing operations are based on the *window* which displays the field, rather than a *string*. The window contains only printable characters, and is filled with blanks. If you want the raw data, you must write your own routine that copies the value out of the buffer and removes the leading and trailing spaces.
- Because editing operations change the content of the buffer to correspond to the window, you should not rely on using buffers for long-term storage of form data.

The function **set_field_status** sets the associated status flag of *field*; **field_status** gets the current value. The status flag is set to a nonzero value whenever the field changes.

The function **set_max_field** sets the maximum size for a dynamic field. An argument of 0 turns off any maximum size threshold for that field.

RETURN VALUE

The **field_buffer** function returns NULL on error. It sets **errno** according to their success:

E_OK
The routine succeeded.

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

The **field_status** function returns **TRUE** or **FALSE**.

The remaining routines return one of the following:

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.

SEE ALSO

curses(3X) and related pages whose names begin "form_" for detailed descriptions of the entry points.

NOTES

The header file **<form.h>** automatically includes the header file

When configured for wide characters, **field_buffer** returns a pointer to temporary storage (allocated and freed by the library). The application should not attempt to modify the data. It will be freed on the next call to **field_buffer** to return the same buffer. **<curses.h>**.

PORTABILITY

These routines emulate the System V forms library. They were not supported on Version 7 or BSD

versions.

The **set_max_field** function checks for an ncurses extension **O_INPUT_FIELD** which allows a dynamic field to shrink if the new limit is smaller than the current field size.

AUTHORS

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