NAME

form_fieldtype - define validation-field types

SYNOPSIS

```
#include <form.h>

FIELDTYPE *new_fieldtype(
   bool (* const field_check)(FIELD *, const void *),
   bool (* const char_check)(int, const void *));
int free_fieldtype(FIELDTYPE *fieldtype);

int set_fieldtype_arg(
   FIELDTYPE *fieldtype,
   void *(* const make_arg)(va_list *),
   void *(* const copy_arg)(const void *),
   void  (* const free_arg)(void *));
int set_fieldtype_choice(
   FIELDTYPE *fieldtype,
   bool (* const next_choice)(FIELD *, const void *),
   bool (* const prev_choice)(FIELD *, const void *));

FIELDTYPE *link_fieldtype(FIELDTYPE *type1,
                          FIELDTYPE *type2);
```

DESCRIPTION

new_fieldtype

The function **new_fieldtype** creates a new field type usable for data validation. Its parameters are function pointers:

field_check

This function checks the validity of an entered data string whenever the user attempts to leave a field. It has two arguments:

- The (FIELD *) argument is passed in so the validation predicate can see the field's buffer, sizes and other attributes.

- The second argument is an argument-block structure, about which more below.

char_check

This function validates input characters as they are entered. The form library passes it the

character to be checked and a pointer to an argument-block structure.

free_fieldtype

The **free_fieldtype** function frees the space allocated for a given validation type by **new_fieldtype**.

set_fieldtype_arg

The function **set_fieldtype_arg** associates three storage-management functions with a field type:

make_arg

This function is automatically applied to the list of arguments you give **set_field_type** when attaching validation to a field. It stores the arguments in an allocated argument-block object which is used when validating input.

copy_arg

This function may be used by applications to copy argument-blocks.

free_arg

Frees an argument-block structure.

You must supply the *make_arg* function. The other two are optional: you may supply NULL for them. In this case, the form library assumes that *make_arg* does not allocate memory but simply loads the argument into a single scalar value.

set_fieldtype_choice

The form driver requests **REQ_NEXT_CHOICE** and **REQ_PREV_CHOICE** assume that the possible values of a field form an ordered set, and provide the forms user with a way to move through the set.

The **set_fieldtype_choice** function allows forms programmers to define successor and predecessor functions for the field type. These functions take the field pointer and an argument-block structure as arguments.

link_fieldtype

The function **link_fieldtype** creates a new field type from the two given types. They are connected by an logical 'OR'.

RETURN VALUE

The pointer-valued routines return NULL on error. They set **errno** according to their success:

E_OK

The routine succeeded.

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

E_SYSTEM_ERROR
System error occurred, e.g., malloc failure.

The integer-valued routines return one of the following codes on error:

E_OK
The routine succeeded.

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

E_CONNECTED
The field is already connected to a form.

E_CURRENT
The field is the current field.

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

SEE ALSO

curses(3X), **form**(3X), **form_field_validation**(3X).

NOTES

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

PORTABILITY

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

AUTHORS

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