NAME

cdk_mentry − curses multiple line entry widget.

SYNOPSIS

cc [*flag* . . .] *file* . . . **−lcdk** [*library* . . .]

#include <cdk.h>

```
char *activateCDKMentry (
                CDKMENTRY *mentry,
                chtype *actions);

void cleanCDKMentry (
                CDKMENTRY *mentry);

void destroyCDKMentry (
                CDKMENTRY *mentry);

void drawCDKMentry (
                CDKMENTRY *mentry,
                boolean box);

void drawCDKMentryField (
                CDKMENTRY *mentry);

void eraseCDKMentry (
                CDKMENTRY *mentry);

boolean getCDKMentryBox (
                CDKMENTRY *mentry);

chtype getCDKMentryFillerChar (
                CDKMENTRY *mentry);

chtype getCDKMentryHiddenChar (
                CDKMENTRY *mentry);

int getCDKMentryMin (
                CDKMENTRY *mentry);

char *getCDKMentryValue (
                CDKMENTRY *mentry);

char *injectCDKMentry (
                CDKMENTRY *mentry,
                chtype input);

void moveCDKMentry (
                CDKMENTRY *mentry,
                int xpos,
                int ypos,
                boolean relative,
                boolean refresh);

CDKMENTRY *newCDKMentry (
                CDKSCREEN *cdkscreen,
                int xpos,
                int ypos,
                const char *title,
                const char *label,
                chtype fieldAttribute,
                chtype fillerCharacter,
                EDisplayType displayType,
                int fieldWidth,
                int fieldRows,
                int logicalRows,
                int minimumLength,
                boolean box,
```

boolean *shadow***);**

void positionCDKMentry (
CDKMENTRY ****mentry*);**

void setCDKMentry (
CDKMENTRY ****mentry*,**
const char ****value*,**
int *minimumLength***,**
boolean *box***);**

void setCDKMentryBackgroundAttrib (
CDKMENTRY ****mentry*,**
chtype *attribute***);**

void setCDKMentryBackgroundColor (
CDKMENTRY ****mentry*,**
const char * *color***);**

void setCDKMentryBox (
CDKMENTRY ****mentry*,**
boolean *boxWidget***);**

void setCDKMentryBoxAttribute (
CDKMENTRY ****mentry*,**
chtype *character***);**

void setCDKMentryCB (
CDKMENTRY ****mentry*,**
MENTRYCB *callBackFunction***);**

void setCDKMentryFillerChar (
CDKMENTRY ****mentry*,**
chtype *filler***);**

void setCDKMentryHiddenChar (
CDKMENTRY ****mentry*,**
chtype *character***);**

void setCDKMentryHorizontalChar (
CDKMENTRY ****mentry*,**
chtype *character***);**

void setCDKMentryLLChar (
CDKMENTRY ****mentry*,**
chtype *character***);**

void setCDKMentryLRChar (
CDKMENTRY ****mentry*,**
chtype *character***);**

void setCDKMentryMin (
CDKMENTRY ****mentry*,**
int *maximum***);**

void setCDKMentryPostProcess (
CDKMENTRY ****mentry*,**
PROCESSFN *callback***,**
void * *data***);**

void setCDKMentryPreProcess (
CDKMENTRY ****mentry*,**
PROCESSFN *callback***,**
void * *data***);**

void setCDKMentryULChar (
CDKMENTRY ****mentry*,**
chtype *character***);**

void setCDKMentryURChar (
    **CDKMENTRY ****mentry*,
    chtype *character***);**

void setCDKMentryValue (
    **CDKMENTRY ****mentry*,
    const char ****value*);**

void setCDKMentryVerticalChar (
    **CDKMENTRY ****mentry*,
    chtype *character***);**

DESCRIPTION

The Cdk mentry widget creates a multiple line entry box with a label and an entry field. The following are functions which create or manipulate the Cdk mentry box widget.

AVAILABLE FUNCTIONS

activateCDKMentry

activates the mentry widget and lets the user interact with the widget.

- The parameter **mentry** is a pointer to a non-NULL mentry widget.
- If the **actions** parameter is passed with a non-NULL value, the characters in the array will be injected into the widget.

  To activate the widget interactively pass in a *NULL* pointer for **actions**.

If the character entered into this widget is *RETURN* or *TAB* then this function will return a *char ** representing the information typed into the widget and the widget data *exitType* will be set to *vNORMAL*.

If the character entered was *ESCAPE* then the function will return *NULL* pointer and the widget data *exitType* is set to *vESCAPE_HIT*.

cleanCDKMentry

clears the information from the field.

destroyCDKMentry

removes the widget from the screen and frees memory the object used.

drawCDKMentry

draws the widget on the screen. If the **box** parameter is true, the widget is drawn with a box.

drawCDKMentryField

redraws the multiple line entry field.

eraseCDKMentry

removes the widget from the screen. This does *NOT* destroy the widget.

getCDKMentryBox

returns true if the widget will be drawn with a box around it.

getCDKMentryFillerChar

returns the character being used to draw unused space in the widget.

getCDKMentryHiddenChar

returns the character being used to draw hidden characters in the widget (obsolete).

getCDKMentryMin

returns the minimum length of a string the widget will allow.

getCDKMentryValue

returns the current value of the widget.

injectCDKMentry

injects a single character into the widget.

- The parameter **mentry** is a pointer to a non-NULL mentry.
- The parameter **character** is the character to inject into the widget.

The return value and side-effect (setting the widget data *exitType*) depend upon the injected character:

RETURN or *TAB*
: the function returns a *char* * representing the information typed into the widget. The widget data *exitType* is set to *vNORMAL*.

ESCAPE
: the function returns a *NULL* pointer. The widget data *exitType* is set to *vESCAPE_HIT*.

Otherwise
: unless modified by preprocessing, postprocessing or key bindings, the function returns a *NULL* pointer.

- The widget data *exitType* is set to *vEARLY_EXIT*.

moveCDKMentry

moves the given widget to the given position.

- The parameters **xpos** and **ypos** are the new position of the widget.

  The parameter **xpos** may be an integer or one of the pre-defined values *TOP*, *BOTTOM*, and *CENTER*.

  The parameter **ypos** may be an integer or one of the pre-defined values *LEFT*, *RIGHT*, and *CENTER*.

- The parameter **relative** states whether the **xpos**/**ypos** pair is a relative move or an absolute move.

  For example, if **xpos** = 1 and **ypos** = 2 and **relative** = **TRUE**, then the widget would move one row down and two columns right.

  If the value of **relative** was **FALSE** then the widget would move to the position (1,2).

  Do not use the values *TOP*, *BOTTOM*, *LEFT*, *RIGHT*, or *CENTER* when **relative** = *TRUE*. (weird things may happen).

- The final parameter **refresh** is a boolean value which states whether the widget will get refreshed after the move.

newCDKMentry

creates a mentry widget and returns a pointer to it. Parameters:

screen
: is the screen you wish this widget to be placed in.

xpos controls the placement of the object along the horizontal axis. It may be an integer or one of the pre-defined values *LEFT*, *RIGHT*, and *CENTER*.

ypos controls the placement of the object along the vertical axis. It may be an integer or one of the pre-defined values *TOP*, *BOTTOM*, and *CENTER*.

title is the string which will be displayed at the top of the widget. The title can be more than one line; just provide a carriage return character at the line break.

label
: is the string which will be displayed in the label of the mentry field.

fieldAttribute
: is the attribute of the characters to be displayed when they are typed in.

filler is the character which is to display in an empty space in the mentry field.

displayType
: tells how the mentry field will behave when a character is entered into the field. The following table outlines valid values for this field and what they mean:

Display_Type	Meaning
vCHAR	Only accepts alphabetic characters.

vLCHAR	Only accepts alphabetic characters. Maps the character to lower case when a character has been accepted.
vUCHAR	Only accepts alphabetic characters. Maps the character to upper case when a character has been accepted.
vHCHAR	Only accepts alphabetic characters. Displays a . when a character has been accepted.
vUHCHAR	Only accepts alphabetic characters. Displays a . and maps the character to upper case when a character has been accepted.
vLHCHAR	Only accepts alphabetic characters. Displays a . and maps the character to lower case when a character has been accepted.
vINT	Only accepts numeric characters.
vHINT	Only accepts numeric characters. Displays a . when a character has been accepted.
vMIXED	Accepts any character types.
vLMIXED	Accepts any character types. Maps the character to lower case when an alphabetic character has been accepted.
vUMIXED	Accepts any character types. Maps the character to upper case when an alphabetic character has been accepted.
vHMIXED	Accepts any character types. Displays a . when a character has been accepted.
vLHMIXED	Accepts any character types. Displays a . and maps the character to lower case when a character has been accepted.
vUHMIXED	Accepts any character types. Displays a . and maps the character to upper case when a character has been accepted.
vVIEWONLY	Uneditable field.

fieldRows and

fieldWidth
control the height and width of the field of the widget. If you provide a value of zero for either of the values, the field in the widget will be made as large as it can both in width and in height. If you provide a negative value, the field will be created the full height or width minus the value provided.

logicalRows
is the number of rows for the mentry field.

minimumLength
is the number of characters which must be entered before the use can exit the mentry field.

callBackFunction
allows the user to override the key processing element of the widget.

box is true if widget should be drawn with a box around it.

shadow
turns the shadow on or off around this widget.

If the widget could not be created then a *NULL* pointer is returned.

positionCDKMentry
allows the user to move the widget around the screen via the cursor/keypad keys. See **cdk_position (3)** for key bindings.

setCDKMentry
lets the programmer modify certain elements of an existing entry widget. The parameter names correspond to the same parameter names listed in the **newCDKMentry** function.

setCDKMentryBackgroundAttrib
sets the background attribute of the widget. The parameter **attribute** is a curses attribute, e.g., A_BOLD.

setCDKMentryBackgroundColor
sets the background color of the widget. The parameter **color** is in the format of the Cdk format strings. See **cdk_display (3)**.

setCDKMentryBox
sets whether the widget will be drawn with a box around it.

setCDKMentryBoxAttribute
function sets the attribute of the box.

setCDKMentryCB
function allows the programmer to set a different widget input handler. The parameter **callbackFunction** is of type *MENTRYCB*. The default function is *CDKMentryCallBack*.

setCDKMentryFillerChar
sets the character to use when drawing unused space in the field.

setCDKMentryHiddenChar
sets the character to use when a hidden character type is used (obsolete).

setCDKMentryHorizontalChar
function sets the horizontal drawing character for the box to the given character.

setCDKMentryLLChar
function sets the lower left hand corner of the widget's box to the given character.

setCDKMentryLRChar
function sets the lower right hand corner of the widget's box to the given character.

setCDKMentryMin
sets the minimum length of the string that the widget will allow.

setCDKMentryPostProcess
allows the user to have the widget call a function after the key has been applied to the widget.

- The parameter **function** is the callback function.
- The parameter **data** points to data passed to the callback function.

To learn more about post-processing see *cdk_process (3)*.

setCDKMentryPreProcess
allows the user to have the widget call a function after a key is hit and before the key is applied to the widget.

- The parameter **function** is the callback function.
- The parameter **data** points to data passed to the callback function.

To learn more about pre-processing see *cdk_process (3)*.

setCDKMentryULChar
sets the upper left hand corner of the widget's box to the given character.

setCDKMentryURChar
sets the upper right hand corner of the widget's box to the given character.

setCDKMentryValue
sets the current value of the widget.

setCDKMentryVerticalChar
sets the vertical drawing character for the box to the given character.

KEY BINDINGS

When the widget is activated there are several default key bindings which will help the user enter or manipulate the information quickly. The following table outlines the keys and their actions for this widget.

Key	**Action**
Left Arrow	Moves the cursor to the left.
CTRL-B	Moves the cursor to the left.
Right Arrow	Moves the cursor to the right.
CTRL-F	Moves the cursor to the right.
Up Arrow	Moves the cursor one row down.
Down Arrow	Moves the cursor one row up.
Delete	Deletes the character at the cursor.
Backspace	Deletes the character before cursor, moves cursor left.
CTRL-V	Pastes whatever is in the paste buffer, into the widget.
CTRL-X	Cuts the contents from the widget and saves a copy in the paste buffer.
CTRL-Y	Copies the contents of the widget into the paste buffer.
CTRL-U	Erases the contents of the widget.
CTRL-A	Moves the cursor to the beginning of the entry field.
CTRL-E	Moves the cursor to the end of the entry field.
CTRL-T	Transposes the character under the cursor with the character to the right.
Return	Exits the widget and returns a *char ** representing the information which was typed into the field. It also sets the widget data *exitType* in the widget pointer to *vNORMAL*.
Tab	Exits the widget and returns a *char ** representing the information which was typed into the field. It also sets the widget data *exitType* in the widget pointer to *vNORMAL*.
Escape	Exits the widget and returns a *NULL* pointer. It also sets the widget data *exitType* to *vESCAPE_HIT*.
Ctrl-L	Refreshes the screen.

SEE ALSO

cdk(3), **cdk_binding**(3), **cdk_display**(3), **cdk_position**(3), **cdk_screen**(3)