NAME

cdk_itemlist − curses itemlist widget.

SYNOPSIS

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

#include <cdk.h>

int activateCDKItemlist (
CDKITEMLIST ****itemlist*,**
chtype ****actions*);**

void destroyCDKItemlist (
CDKITEMLIST ****itemlist*);**

void drawCDKItemlist (
CDKITEMLIST ****itemlist*,**
boolean *box***);**

void drawCDKItemlistField (
CDKITEMLIST ****itemlist*,**
boolean *highlight***);**

void eraseCDKItemlist (
CDKITEMLIST ****itemlist*);**

boolean getCDKItemlistBox (
CDKITEMLIST ****itemlist*);**

int getCDKItemlistCurrentItem (
CDKITEMLIST ****itemlist*);**

int getCDKItemlistDefaultItem (
CDKITEMLIST ****itemlist*);**

chtype **getCDKItemlistValues (
CDKITEMLIST ****itemlist*,**
int ****listSize*);**

int injectCDKItemlist (
CDKITEMLIST ****itemlist*,**
chtype *input***);**

void moveCDKItemlist (
CDKITEMLIST ****itemlist*,**
int *xpos***,**
int *ypos***,**
boolean *relative***,**
boolean *refresh***);**

CDKITEMLIST *newCDKItemlist (
CDKSCREEN ****cdkscreen*,**
int *xpos***,**
int *ypos***,**
char ****title*,**
char ****label*,**
char *****itemList*,**
int *itemCount***,**
int *defaultItem***,**
boolean *box***,**
boolean *shadow***);**

void positionCDKItemlist (
CDKITEMLIST ****itemlist*);**

void setCDKItemlist (
CDKITEMLIST ****itemlist*,**
char *****itemList*,**

int *itemCount*,
int *currentSelection*,
boolean *box*);

void setCDKItemlistBackgroundAttrib (
**CDKITEMLIST ***_itemlist_,
chtype*attribute*);

void setCDKItemlistBackgroundColor (
**CDKITEMLIST ***_itemlist_,
**char ***_color_);

void setCDKItemlistBox (
**CDKITEMLIST ***_itemlist_,
boolean *box*);

void setCDKItemlistBoxAttribute (
**CDKITEMLIST ***_itemlist_,
chtype *character*);

void setCDKItemlistCurrentItem (
**CDKITEMLIST ***_itemlist_,
int *currentItem*);

void setCDKItemlistDefaultItem (
**CDKITEMLIST ***_itemlist_,
int *defaultItem*);

void setCDKItemlistHorizontalChar (
**CDKITEMLIST ***_itemlist_,
chtype *character*);

void setCDKItemlistLLChar (
**CDKITEMLIST ***_itemlist_,
chtype *character*);

void setCDKItemlistLRChar (
**CDKITEMLIST ***_itemlist_,
chtype *character*);

void setCDKItemlistPostProcess (
**CDKITEMLIST ***_itemlist_,
PROCESSFN *callback*,
**void ***_data_);

void setCDKItemlistPreProcess (
**CDKITEMLIST ***_itemlist_,
PROCESSFN *callback*,
**void ***_data_);

void setCDKItemlistULChar (
**CDKITEMLIST ***_itemlist_,
chtype *character*);

void setCDKItemlistURChar (
**CDKITEMLIST ***_itemlist_,
chtype *character*);

void setCDKItemlistValues (
**CDKITEMLIST ***_itemlist_,
**char ****_itemList_,
int *itemCount*);

void setCDKItemlistVerticalChar (
**CDKITEMLIST ***_itemlist_,
chtype *character*);

DESCRIPTION

The Cdk itemlist widget creates a widget which allows a user to select from a list of preset character strings such as the days of the week or the months of the year. The following functions create or manipulate the Cdk itemlist widget.

AVAILABLE FUNCTIONS

activateCDKItemlist

activates the itemlist widget and lets the user interact with the widget. The parameter **itemlist** is a pointer to a non-NULL itemlist 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 value from 0 to the number of buttons -1, representing the button selected. It will also set the widget data *exitType* to *vNORMAL*. If the character entered into this widget was *ESCAPE* then the widget will return a -1 and the widget data *exitType* will be set to *vESCAPE_HIT*.

destroyCDKItemlist

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

drawCDKItemlist

draws the itemlist widget on the screen. The **box** option is true if the widget is drawn with a box.

drawCDKItemlistField

draws the contents of the field.

eraseCDKItemlist

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

getCDKItemlistBox

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

getCDKItemlistCurrentItem

returns the index of the currently displayed item in the widget.

getCDKItemlistDefaultItem

returns the index of the default item in the widget.

getCDKItemlistValues

returns the list of pointers to the items. The parameter **size** points to a location which receives the item count.

injectCDKItemlist

injects a single character into the widget. The parameter **itemlist** is a pointer to a non-NULL itemlist widget. 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 value ranging from zero to one less than the number of buttons, representing the button selected. The widget data *exitType* is set to *vNORMAL*.

ESCAPE

the function returns -1. The widget data *exitType* is set to *vESCAPE_HIT*.

Otherwise

unless modified by preprocessing, postprocessing or key bindings, the function returns -1. The widget data *exitType* is set to *vEARLY_EXIT*.

moveCDKItemlist

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.

newCDKItemlist
creates a pointer to an itemlist widget. 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 to use as the label of the itemlist field.

itemList
is the list of the strings which will be displayed in the widget.

itemCount
is the number of elements in the list.

defaultItem
is the index of the default item for the list.

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.

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

setCDKItemlist
lets the programmer modify certain elements of an existing itemlist widget. The parameter names correspond to the same parameter names listed in the **newCDKItemlist** function.

setCDKItemlistBackgroundAttrib
the background color attribute the widget. The parameter **attribute** is a curses attribute, e.g., A_BOLD.

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

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

setCDKItemlistBoxAttribute
sets the attribute of the box.

setCDKItemlistCurrentItem
sets the currently displayed item in the widget.

setCDKItemlistDefaultItem
sets the default item in the widget.

setCDKItemlistHorizontalChar
sets the horizontal drawing character for the box to the given character.

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

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

setCDKItemlistPostProcess
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)*.

setCDKItemlistPreProcess
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)*.

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

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

setCDKItemlistValues
sets the contents of the list from an array of string pointers **item** whose final index is given by **count**. If **defaultItem** is in the range 0..**count**, that sets the default item value for the list.

setCDKItemlistVerticalChar
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 Down Arrow - p	Shift the list one column to the left.
Right Arrow Up Arrow Space + n	Shift the list one column to the right.
d D	Display the default item.
0 $	Display the first item in the list. Display the last item in the list.
Return	Exit the widget and return an integer representing the current selection. Also set the widget data *exitType* to *vNORMAL*.
Tab	Exit the widget and return an integer representing the current selection. Also set the widget data *exitType* to *vNORMAL*.
Escape	Exit the widget and return -1. Also set 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_process**(3), **cdk_screen**(3)