NAME

cdk_traverse - functions to support keyboard traversal

SYNOPSIS

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

#include <cdk.h>

CDKOBJS *getCDKFocusCurrent (CDKSCREEN ***screen*);**

CDKOBJS *setCDKFocusCurrent (CDKSCREEN ***screen*, CDKOBJS ***obj***);**

CDKOBJS *setCDKFocusFirst (CDKSCREEN ***screen*);**

CDKOBJS *setCDKFocusLast (CDKSCREEN ***screen*);**

CDKOBJS *setCDKFocusNext (CDKSCREEN ***screen*);**

CDKOBJS *setCDKFocusPrevious (CDKSCREEN ***screen*);**

int traverseCDKScreen (CDKSCREEN ***screen*);**

void exitOKCDKScreen (CDKSCREEN ***screen*);**

void exitCancelCDKScreen (CDKSCREEN ***screen*);**

void resetCDKScreen (CDKSCREEN ***screen*);**

void exitOKCDKScreenOf(CDKOBJS ***obj*);**

void exitCancelCDKScreenOf (CDKOBJS ***obj*);**

void resetCDKScreenOf (CDKOBJS ***obj*);**

void traverseCDKOnce (
CDKSCREEN ***screen*,**
CDKOBJS ***curobj*,**
int *keyCode***,**
boolean *functionKey***,**
CHECK_KEYCODE *funcMenuKey***);**

DESCRIPTION

The functions above handle the traversal of a screen populated with various widgets. Once the screen has been created and populated with widgets, a single call to **traverseCDKScreen()** will allow the user to move between widgets and enter data (or otherwise manipulate widgets). Other functions are provided for use as callbacks by the widgets on the screen. Finally, there are several functions which allow the caller to manipulate the state of the traversal, i.e., the object which has focus.

In order for widgets to be used on a screen which is to be handled by **traverseCDKScreen()**, it must have the following methods available:

```
injectCharObj
inputWindowObj
focusObj
unfocusObj
saveDataObj
refreshDataObj
```

In addition, the following object properties must be properly handled:

```
acceptsFocus
hasFocus
inputWindow
dataPtr
dataType
```

At the time of this writing, not all widgets have been modified to work with the screen-traversal facility.

AVAILABLE FUNCTIONS

int traverseCDKScreen (CDKSCREEN ***screen*);**

This function contains the main screen traversal engine. It does the following:

1. Calls the refreshData method on each of the widgets to tell them to update their appearance to match the data which are referenced by their respective data pointers.

2. Calls the focusObject method on the first widget.
3. Repeats the following until one of the exit functions listed above has been called:
   * Read a keystroke from the keyboard.
   * If the keystroke is ESCAPE and a menu widget is present, activate the menu and traverse it until the user selects an entry or hits TAB.
   * If the keystroke is TAB/BACKTAB then call the unfocusObject method on the current widget, and move focus to the next/previous widget (not counting menu widgets). Call the focusObject method on the newly current widget.
   * If the keystroke is the EXIT-SAVE keystroke, then call the saveData method on each widget and return 1.
   * If the keystroke is the EXIT-CANCEL keystroke, return 0 without saving changes made by the user.
   * If the keystroke is the RESET-DATA keystroke, then call the refreshData method on each of the widgets to reset their appearance to match the data values that were present upon entry.
   * Otherwise, pass the keystroke to the current widget.

CDKOBJS *getCDKFocusCurrent (CDKSCREEN *_screen_);
Return a pointer to the object which currently has focus in the given screen.

CDKOBJS *setCDKFocusCurrent (CDKSCREEN *_screen_, CDKOBJS *_obj_);
Set the focus to the given object, if the screen contains that object. If the screen does not contain the object, return null. Otherwise, return the object.

CDKOBJS *setCDKFocusFirst (CDKSCREEN *_screen_);
Set focus on the first object in the given screen.

CDKOBJS *setCDKFocusLast (CDKSCREEN *_screen_);
Set focus on the last object in the given screen.

CDKOBJS *setCDKFocusNext (CDKSCREEN *_screen_);
Set focus on the next object in the given screen.

CDKOBJS *setCDKFocusPrevious (CDKSCREEN *_screen_);
Set focus on the previous object in the given screen.

exitOKCDKScreen
Causes the traversal engine to exit after calling the saveData method for each of the widgets.

exitOKCDKScreenOf
Calls **exitOKCDKScreen()** on the screen associated with widget *obj*. This function was designed to be used as a callback routine for a button widget used as an OK button on a data-entry screen.

exitCancelCDKScreen
Causes the traversal engine to exit without saving user modified data.

exitCancelCDKScreenOf
Calls **exitCancelCDKScreen()** on the screen associated with widget *obj*. This function was designed to be used as a callback routine for a button widget used as a Cancel button on a data-entry screen.

resetCDKScreen
Causes the traversal engine to call the refreshData method for each widget. This will cause any unsaved changes to be discarded and the widget states will be restored to their initial values.

resetCDKScreenOf
Calls **resetCDKScreen()** on the screen associated with widget *obj*. This function was designed to be used as a callback routine for a button widget used as a Reset button on a data-entry screen.

traverseCDKOnce
This is a utility function, one of the pieces from which you can construct a customized version of **traverseCDKScreen**.

BUGS

Not all widgets have had the extra methods added so that they work with the screen traversal engine.

AUTHOR

Grant Edwards, Aspen Research Corporation
Thomas Dickey and contributors.

SEE ALSO

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