NAME

curses − screen/window management library

SYNOPSIS

cc demo.c -lcurses

DESCRIPTION

Curses is a library of screen and window management routines. It is modeled after the UNIX curses and ncurses libraries. Normally, programs written for curses should be easily ported to UNIX, and vice versa.

To use the routines, the function initscr() must first be called. This creates two 'windows' for the user: stdscr and curscr. Stdscr is the default window for the user to make changes on, and curscr reflects the current contents of the physical display screen. The user writes or edits the stdscr window to his liking, then calls the refresh() function to make curscr and the physical screen look like stdscr. When the user program terminates, it should call the endwin() function to restore things to normal.

There are all sorts of window manipulation routines available to the programmer: auxiliary windows may be created, edited, moved and deleted. The terminal may be set in many different modes, output text may be attributed with blink, blank, bold and reverse attributes. Screen colors may also be set, foreground and background. There are window-specific printf- and scanf-like routines, routines for scrolling, box-drawing, window overlaying, clearing routines etc.

For more and detailed information, see the library source codes. All curses functions are preceded by a complete description.

FUNCTIONS

Below is a list over the available functions, together with a brief description of what they do. In general, functions whose names start with 'w' differ from the one without 'w' (like wmove vs. move) signify that a specific window is used. Without a 'w', sdtscr is implied. The functions that start with 'mv' before the 'genereic' function name signify that a cursor motion should be made before the actual work. 'mv' and 'w' combine as expected.

Most routines that return an int will return the manifest constant ERR if there is a failure during execution. Routines that return a char actually return an int, so that ERR does not conflict with the character code 0xff. All characters from 0 to 0xff are allowed for usage with curses.

Some routines, like {mv}{w} printw() and {mv}{w}scanw() return a meaningful positive value if the operation is successful.

The curses package uses some predefined types, variables and manifest constants that are also available to the programmer. There are also a few globally accessible variables that should not be touched by the application program. Those untouchable variables have names starting with an underscore (_) to avoid conflicts. The user-accessible types, variables and constants are (there are a number of other constants defining character attribute names and function key names - consult <curses.h> for details):

(manifest constants)

TRUE	boolean true
FALSE	boolean false
ERR	unsuccessfull operation
OK	successfull operation

(types)

WINDOW	a window structure type
bool	boolean flag type

(variables)

WINDOW curscr	physical display image
WINDOW stdscr	default user drawing board
int LINES	terminal height
int COLS	terminal width

int NONL	\n causes CR and LF when TRUE

The following is an alphabetical list of the curses functions, together with their types, parameters and a short comment for each (win is a window, ch, vc, hc are characters, buf is a character buffer, attrs is an attribute bit map, bf is a boolean flag. Note that 'characters' in this context usually can have 16 bits):

int waddch(win,ch) int addch(ch) int mvaddch(y,x,ch) int mvwaddch(win,y,x,ch)	put char in stdscr
int waddstr(win,str) int addstr(str) int mvaddstr(y,x,str) int mvwaddstr(win,y,x,str)	put string in stdscr
void wattroff(win,attrs) void attroff(attrs)	clear attribute(s) in window
void wattron(win,attrs) void attron(attrs)	add attribute(s) in window
void wattrset(win,attrs) void attrset(attrs)	set window char attributes
int baudrate()	dummy for compatibility
void beep()	ring the bell or visible bell if no bell available
void flash()	flash terminal screen or rings bell if no visible bell available
void wbox(win,miny,minx,maxy,maxx,vc,hc) void box(win,vc,hc)	box in a window, with given characters
void cbreak()	set terminal cbreak mode
void wclear(win) void clear()	clear stdscr
void clearok(win,bf)	marks window for screen clear
int wclrtobot(win) int clrtobot() int mvclrtoeol(y,x) int mvwclrtobot(win,y,x)	clear from cursor to end of line and all lines down this line
int wclrtoeol(win) int clrtoeol() int mvclrtoeol(y,x) int mvwclrtoeol(win,y,x)	clear from cursor to end of line
int wdelch(win) int delch() int mvdelch(y,x) int mvwdelch(win,y,x)	delete a char in a window

int wdeleteln(win)	delete a line in a window
int deleteln()	
int mvdeleteln(y,x)	
int mvwdeleteln(win,y,x)	
void delwin(win)	delete a window or a subwindow
void doupdate()	update physical screen
void echo()	set terminal echo mode
int endwin()	cleanup and curses finitialization
void werase(win)	erase a window
void erase()	
int erasechar()	return char delete character
int fixterm()	dummy for compatibility
void flushinp()	kill pending keyboard input
int wgetch(win)	get char via a window
int getch()	
int mvgetch(y,x)	
int mvwgetch(win,y,x)	
int wgetstr(win,str)	get string via window to a buffer
int getstr(str)	
int mvgetstr(y,x,str)	
int mvwgetstr(win,y,x,str)	
void getyx(win,y,x)	get a window's cursor position
int gettmode()	dummy for compatibility
void idlok(win,bf)	dummy for compatibility
WINDOW *initscr()	curses initialization (ret stdscr or NULL)
int winch(win)	get char at window cursor
int inch()	
int mvinch(y,x)	
int mvwinch(win,y,x)	
int winsch(win,ch)	insert character in a window
int insch(ch)	
int mvinsch(y,x,ch)	
int mvwinsch(win,y,x,ch)	
int winsertln(win)	insert new line in a window
int insertln()	
int mvinsertln(y,x)	
int mvwinsertln(win,y,x)	
void keypad(win,bf)	marks a window for keypad usage
int killchar()	return line delete character
char *longname()	returns terminal description string
void leaveok(win,bf)	marks window for cursor 'update leave'
void meta(win,bf)	marks window for meta
int move(y,x)	move cursor in stdscr
int mvcur(oldy,oldx,y,x)	move terminal cursor to <y,x>

int mvprintw(y,x,fmt,args)	move & print string in stdscr
int mvscanw(y,x,fmt,args)	move & get values via stdscr
int mvwin(win,y,x)	move window on physical screen
int mvwprintw(win,x,y,fmt,args)	move & print string in a window
int mvwscanw(win,y,x,fmt,args)	move & get values via a window
WINDOW *newwin(lines,cols,begy,begx)	create a new window
void nl()	set terminal cr-crlf mapping mode
void nocbreak()	unset terminal cbreak mod
void nodelay(win,bf)	marks window for no input wait
void noecho()	unset terminal echo mode
void nonl()	unset terminal cr-crlf mapping mode
void noraw()	unset raw terminal mode
void overlay(win1,win2)	overlay one window on another
void overwrite(win1,win2)	overwrite one window on another
int printw(fmt,args)	print string in stdscr
void raw()	set raw terminal mode
void refrbrk(bf)	set screen update break mode
void refresh()	refresh stdscr
int resetterm()	dummy for compatibility
int resetty()	restore terminal I/O modes
int saveoldterm()	dummy for compatibility
int saveterm()	dummy for compatibility
int savetty()	save terminal I/O modes
int scanw(fmt,args)	get values via stdscr
void scroll(win)	scroll scrolling region of a window
void scrollok(win,bf)	marks a window to allow scroll
void setcolors(A_COLOR(for,back))	sets the forground and background colors of stdscr
void set_curs(visibility)	0 for invisible, 1 for visible, 2 for good visible
int setsrcreg(miny,maxy)	define stdscr's scroll region
int setterm()	dummy for compatibility
int setupterm(term,fd,errret)	set up terminal
void standend()	start normal chars in stdscr
void standout()	start standout chars in stdscr
WINDOW *subwin(win,lines,cols,begy,begx)	create a sub-window in window win
int tabsize(ts)	set/get tabsize of stdscr
void touchwin(win)	mark a window as totally modified
char *unctrl(ch)	char-to-string converter
int wmove(win,y,x)	move cursor in a window
void wnoutrefresh(win)	create internal screen image
int wprintw(win,fmt,args)	print string in a window
void wrefresh(win)	refresh window
int wscanw(win,fmt,args)	get values via a window
void wsetcolors(win,A_COLOR(for,back))	sets the forground and background colors of the specified window
int wsetsrcreg(win,miny,maxy)	define a window's scrolling region
void wstandend(win)	start normal chars in window
void wstandout(win)	start standout chars in window
int wtabsize(win,ts)	set/get tabsize of a window

BUGS

Function keys are not available under the MINIX version.