SunOS リファレンスマニュアル 3 : Curses ライブラリ関数

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 816–3990–10
2002 年 5 月

原典: *man pages section 3 : Curses Library Functions*

Part No: 816-0215-10

Revision A

Adobe PostScript

020325@2851

目次

はじめに　**7**

SunOS リファレンスマニュアル **3 : Curses** ライブラリ関数　**11**
addnwstr(3CURSES)　12
addwch(3CURSES)　13
addwchnstr(3CURSES)　16
addwchstr(3CURSES)　18
addwstr(3CURSES)　20
adjcurspos(3CURSES)　21
curses(3CURSES)　22
curs_addwch(3CURSES)　37
curs_addwchstr(3CURSES)　40
curs_addwstr(3CURSES)　42
curs_alecompat(3CURSES)　43
curs_getwch(3CURSES)　44
curs_getwstr(3CURSES)　49
curs_inswch(3CURSES)　50
curs_inswstr(3CURSES)　51
curs_inwch(3CURSES)　53
curs_inwchstr(3CURSES)　54
curs_inwstr(3CURSES)　55
curs_pad(3CURSES)　56
echowchar(3CURSES)　58
getnwstr(3CURSES)　61
getwch(3CURSES)　62
getwstr(3CURSES)　67

innwstr(3CURSES) 68
insnwstr(3CURSES) 69
inswch(3CURSES) 71
inswstr(3CURSES) 72
inwch(3CURSES) 74
inwchnstr(3CURSES) 75
inwchstr(3CURSES) 76
inwstr(3CURSES) 77
movenextch(3CURSES) 78
moveprevch(3CURSES) 79
mvaddnwstr(3CURSES) 80
mvaddwch(3CURSES) 81
mvaddwchnstr(3CURSES) 84
mvaddwchstr(3CURSES) 86
mvaddwstr(3CURSES) 88
mvgetnwstr(3CURSES) 89
mvgetwch(3CURSES) 90
mvgetwstr(3CURSES) 95
mvinnwstr(3CURSES) 96
mvinsnwstr(3CURSES) 97
mvinswch(3CURSES) 99
mvinswstr(3CURSES) 100
mvinwch(3CURSES) 102
mvinwchnstr(3CURSES) 103
mvinwchstr(3CURSES) 104
mvinwstr(3CURSES) 105
mvwaddnwstr(3CURSES) 106
mvwaddwch(3CURSES) 107
mvwaddwchnstr(3CURSES) 110
mvwaddwchstr(3CURSES) 112
mvwaddwstr(3CURSES) 114
mvwgetnwstr(3CURSES) 115
mvwgetwch(3CURSES) 116
mvwgetwstr(3CURSES) 121
mvwinnwstr(3CURSES) 122
mvwinsnwstr(3CURSES) 123
mvwinswch(3CURSES) 125
mvwinswstr(3CURSES) 126

mvwinwch(3CURSES) 128
mvwinwchnstr(3CURSES) 129
mvwinwchstr(3CURSES) 130
mvwinwstr(3CURSES) 131
newpad(3CURSES) 132
pechochar(3CURSES) 134
pechowchar(3CURSES) 136
pnoutrefresh(3CURSES) 138
prefresh(3CURSES) 140
subpad(3CURSES) 142
ungetwch(3CURSES) 144
waddnwstr(3CURSES) 149
waddwch(3CURSES) 150
waddwchnstr(3CURSES) 153
waddwchstr(3CURSES) 155
waddwstr(3CURSES) 157
wadjcurspos(3CURSES) 158
wechowchar(3CURSES) 159
wgetnwstr(3CURSES) 162
wgetwch(3CURSES) 163
wgetwstr(3CURSES) 168
winnwstr(3CURSES) 169
winsnwstr(3CURSES) 170
winswch(3CURSES) 172
winswstr(3CURSES) 173
winwch(3CURSES) 175
winwchnstr(3CURSES) 176
winwchstr(3CURSES) 177
winwstr(3CURSES) 178
wmovenextch(3CURSES) 179
wmoveprevch(3CURSES) 180

はじめに

概要

SunOS リファレンスマニュアルは、初めて SunOS を使用するユーザーやすでにある程度の知識を持っているユーザーのどちらでも対応できるように解説されています。このマニュアルを構成するマニュアルページは一般に参照マニュアルとして作られており、チュートリアルな要素は含んでいません。それぞれのコマンドを実行すると、どのような結果が得られるかについて、詳しく説明されています。なお、各マニュアルページの内容はオンラインでも参照することができます。

このマニュアルは、マニュアルページの内容によっていくつかのセクションに分かれています。各セクションについて以下に簡単に説明します。

- セクション 1 は、オペレーティングシステムで使えるコマンドを説明します。
- セクション 1M は、システム保守や管理用として主に使われるコマンドを説明します。
- セクション 2 は、すべてのシステムコールについて説明します。ほとんどのシステムコールに 1 つまたは複数のエラーがあります。エラーの場合、通常ありえない戻り値が返されます。
- セクション 3 は、さまざまなライブラリ中の関数について説明します。ただし、UNIX システムプリミティブを直接呼び出す関数については、セクション 2 で説明しています。
- セクション 5 は、文字セットテーブルなど他のセクションには該当しないものについて説明します。

以下に、このマニュアルの項目を表記されている順に説明します。ほとんどのマニュアルページが下記の項目からなる共通の書式で書かれていますが、必要でない項目については省略されています。たとえば、記述すべきバグがコマンドにない場合などは、「使用上の留意点」という項目はありません。各マニュアルページの詳細は各セクションの `intro` を、マニュアルページの一般的な情報については `man`(1) を参照してください。

名前　コマンドや関数の名称と概略が示されています。

形式　コマンドや関数の構文が示されています。標準パスにコマンドやファイルが存在しない場合は、フルパス名が示されます。字体は、コマンド、オプションなどの定数にはボールド体 (bold) を、引数、パラメータ、置換文字などの変数にはイタリック体 (Italic) または <日本語訳> を使用しています。オプションと引数の順番は、アルファベット順です。特別な指定が必要な場合を除いて、1 文字の引数、引数のついたオプションの順に書かれています。

以下の文字がそれぞれの項目で使われています。

- [ ]　このかっこに囲まれたオプションや引数は省略できます。このかっこが付いていない場合には、引数を必ず指定する必要があります。
- . . .　省略符号。前の引数に変数を付けたり、引数を複数指定したりできることを意味します (例: ‘ `filename` . . . ’)。
- |　区切り文字 (セパレータ)。この文字で分割されている引数のうち 1 つだけを指定できます。
- { }　この大かっこに囲まれた複数のオプションや引数は省略できます。かっこ内を 1 組として扱います。

プロトコル　この項が使われているのは、プロトコルが記述されているファイルを示すサブセクション 3R だけです。パス名は常にボールド体 (bold) で示されています。

機能説明　コマンドの機能とその動作について説明します。実行時の詳細を説明していますが、オプションの説明や使用例はここでは示されていません。対話形式のコマンド、サブコマンド、リクエスト、マクロ、関数などに関しては「使用法」で説明します。

IOCTL　セクション 7 だけに使用される項です。`ioctl`(2) システムコールへのパラメータは `ioctl` と呼ばれ、適切なパラメータを持つデバイスクラスのマニュアルページだけに記載されています。特定のデバイスに関する `ioctl` は、(そのデバイスのマニュアルページに) アルファベット順に記述されています。デバイスの特定のクラスに関する `ioctl` は、`mtio`(7I) のように `io` で終わる名前が付いているデバイスクラスのマニュアルページに記載されています。

オプション　各オプションがどのように実行されるかを説明しています。「形式」で示されている順に記述されています。オプションの引数はこの項目で説明され、必要な場合はデフォルト値を示します。

オペランド　コマンドのオペランドを一覧表示し、各オペランドがコマンドの動作にどのように影響を及ぼすかを説明しています。

出力　コマンドによって生成される出力 (標準出力、標準エラー、または出力ファイル) を説明しています。

戻り値	値を返す関数の場合、その値を示し、値が返される時の条件を説明しています。関数が 0 や -1 のような一定の値だけを返す場合は、値と説明の形で示され、その他の場合は各関数の戻り値について簡単に説明しています。void として宣言された関数はこの項では扱いません。
エラー	失敗の場合、ほとんどの関数はその理由を示すエラーコードを `errno` 変数の中に設定します。この項ではエラーコードをアルファベット順に記述し、各エラーの原因となる条件について説明します。同じエラーの原因となる条件が複数ある場合は、エラーコードの下にそれぞれの条件を別々のパラグラフで説明しています。
使用法	この項では、使用する際の手がかりとなる説明が示されています。特定の決まりや機能、詳しい説明の必要なコマンドなどが示されています。組み込み機能については、以下の小項目で説明しています。 コマンド 修飾子 変数 式 入力文法
使用例	コマンドや関数の使用例または使用方法を説明しています。できるだけ実際に入力するコマンド行とスクリーンに表示される内容を例にしています。例の中には必ず `example%` のプロンプトが出てきます。スーパーユーザーの場合は `example#` のプロンプトになります。例では、その説明、変数置換の方法、戻り値が示され、それらのほとんどが「形式」、「機能説明」、「オプション」、「使用法」の項からの実例となっています。
環境	コマンドや関数が影響を与える環境変数を記述し、その影響について簡単に説明しています。
終了ステータス	コマンドが呼び出しプログラムまたはシェルに返す値と、その状態を説明しています。通常、正常終了には 0 が返され、0 以外の値はそれぞれのエラー状態を示します。
ファイル	マニュアルページが参照するファイル、関連ファイル、およびコマンドが作成または必要とするファイルを示し、各ファイルについて簡単に説明しています。
属性	属性タイプとその対応する値を定義することにより、コマンド、ユーティリティ、およびデバイスドライバの特性を一覧しています。詳細は `attributes`(5) を参照してください。
関連項目	関連するマニュアルページ、当社のマニュアル、および一般の出版物が示されています。

診断	エラーの発生状況と診断メッセージが示されています。メッセージはボールド体 (bold) で、変数はイタリック体 (Italic) または <日本語訳> で示されており、C ロケール時の表示形式です。
警告	作業に支障を与えるような現象について説明しています。診断メッセージではありません。
注意事項	それぞれの項に該当しない追加情報が示されています。マニュアルページの内容とは直接関係のない事柄も参照用に扱っています。ここでは重要な情報については説明していません。
使用上の留意点	すでに発見されているバグについて説明しています。可能な場合は対処法も示しています。

SunOS リファレンスマニュアル 3 : Curses ライブラリ関数

名前

curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_addwch, addwch, waddwch, mvaddwch, mvwaddwch, echowchar, wechowchar – wchar_t 文字を属性とともに curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwch(chtype wch);

int waddwch(WINDOW *win, chtype wch);

int mvaddwch(int y, int x, chtype wch);

int mvwaddwch(WINDOW *win, int y, int x, chtype wch);

int echowchar(chtype wch);

int wechowchar(WINDOW *win, chtype wch);
```

機能説明

addwch()、waddwch()、mvaddwch()、mvwaddwch() の各ルーチンは、wchar_t を保ちながら *wch* 文字をウィンドウ中の現カーソル位置に書き出し、ウィンドウカーソルの位置を進めます。この機能は C 複数バイトライブラリ内にある putwchar(3C) の機能に似ています。右側のマージンでは自動的に復帰改行 (newline) が行われます。scrollok が有効であれば、スクロール領域が上に 1 行スクロールされます。

wch がタブ、復帰改行、バックスペースのいずれかの場合、それに従ってウィンドウ内でカーソルが移動します。なお復帰改行の場合には、カーソル移動に先だって clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wch* が他の制御文字の場合、^*X* 形式で描かれます。制御文字を追加した後で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

ビデオ属性は、論理和をパラメータ中にとることにより wchar_t 文字と組み合わせることができます。その結果、属性の設定も行われます。これが意味することは、inwch() と addwch() を使ってテキスト (属性も含む) をある場所から他の場所へコピーすることが可能であるということです。standout(3CURSES) の、定義済みビデオ属性定数の項を参照してください。

機能的に見ると、echowchar() ルーチンは addwch() と refresh(3CURSES) を連続して呼び出すことと同等で、wechowchar() ルーチンは waddwch() と wrefresh(3CURSES) を連続して呼び出すことと同等です。両ルーチンは 1 文字だけしか出力されないことが判っていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。

線グラフィック

addwch() ファミリのルーチンとともに以下の変数を使用して、線を描画する文字を画面に書き出すことができます。端末に変数が定義されているとき、A_ALTCHARSET ビットがオンになります (curs_attr(3CURSES) の項を参照)。変数が定義されていなければ、以下に述べるデフォルト文字が変数に与えられます。なお各変数の名前は VT100 命名規約に準拠しています。

名前	デフォルト	グリフ記述
ACS_ULCORNER	+	左上角
ACS_LLCORNER	+	左下角
ACS_URCORNER	+	右上角
ACS_LRCORNER	+	右下角
ACS_RTEE	+	右端
ACS_LTEE	+	左端
ACS_BTEE	+	下端
ACS_TTEE	+	上端
ACS_HLINE	–	横線
ACS_VLINE	|	縦線
ACS_PLUS	+	正符号
ACS_S1	-	スキャンライン 1
ACS_S9	_	スキャンライン 9
ACS_DIAMOND	+	ダイヤモンド
ACS_CKBOARD	:	市松模様 (点描)
ACS_DEGREE	'	度数記号
ACS_PLMINUS	#	プラス／マイナス
ACS_BULLET	o	丸
ACS_LARROW	<	左向きの矢印
ACS_RARROW	>	右向きの矢印
ACS_DARROW	v	下向きの矢印
ACS_UARROW	^	上向きの矢印
ACS_BOARD	#	正方形のボード
ACS_LANTERN	#	ランタン記号
ACS_BLOCK	#	正方形のブロック

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 putwchar(3C), clrtoeol(3CURSES), curses(3CURSES), curs_attr(3CURSES), curs_inwch(3CURSES), curs_outopts(3CURSES), refresh(3CURSES), standout(3CURSES), winwch(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

addwch()、mvaddwch()、mvwaddwch()、echowchar() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明

上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwchnstr() 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明

上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwchnstr() 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_alecompat, movenextch, wmovenextch, moveprevch, wmoveprevch, adjcurspos, wadjcurspos – カーソルを文字単位に移動するために ALE curses ライブラリに追加された関数

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int movenextch(void);

int wmovenextch(WINDOW *win);

int moveprevch(void);

int wmoveprevch(WINDOW *win);

int adjcurspos(void);

int wadjcurspos(WINDOW *win);
```

機能説明

上記 `movenextch()` および `wmovenextch()` ルーチンは、カーソルを次の (右方の) 文字に移動します。次の文字が複数カラム文字であれば、新カーソル位置はその文字の最初の (左端の) カラムとなります。現カーソル位置が複数カラム文字の左端カラムであっても、新しい位置は次の文字上に移動します。なお現カーソル位置が複数カラム文字上の場合、単純なカーソル移動 (++x) を実行しても次の文字に移動するとは保証できません。新しいカーソル位置は `getyx`(3CURSES) を実行することにより得られます。

`moveprevc()` および `wmoveprevch()` ルーチンは、上記 `movenextch()` と `wmovenextch()` とは反対方向に、つまり 1 つ前の文字の左端カラムにカーソルを移動します。

`adjcurspos()` および `wadjcurspos()` ルーチンは、現在カーソルが置かれている複数カラム文字の左端のカラムへカーソルを移動します。現在の位置がすでに左端カラムである場合、および現在置かれている文字がシングルカラム文字の場合、これらのルーチンは意味を持たず、カーソル位置は変わりません。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性

次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

`curses`(3CURSES), `getyx`(3CURSES), `attributes`(5)

注意事項

ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`movenextch()`、`moveprevch()`、`adjcurspos()` の各ルーチンはマクロにすることも可能です。

名前 | curses – CRT 画面操作および最適化用パッケージ

形式 |
```
cc [ flag... ] file...- lcurses [ library... ]
#include <curses.h>
```

機能説明 | curses ライブラリ中の各ルーチンは、一定の最適化を行いながら文字画面を更新するための、端末に依存しない方法を提供しています。

curses パッケージを利用すると次のことが可能になります。画面全体のウィンドウ、およびパッドの操作。ウィンドウおよびパッドへの出力。端末の制御と curses 入力／出力オプションの制御。環境問い合わせルーチン。カラー処理。ソフトラベルキーの使用。terminfo のアクセス。低レベル curses ルーチンのアクセス。

curses ライブラリルーチンを初期化するには、ウィンドウや画面を扱うルーチンを使用する前に initscr() または newterm() のルーチンを呼び出さなければなりません。また処理を終了する前に endwin() ルーチンを呼び出さなければなりません。会話型かつ画面指向のプログラムは、エコーなしの「一度に 1 文字」入力を求めるものが大多数ですが、これには次のシーケンスが必要です。

```
initscr,cbreak,noecho;
```

また、多くのプログラムはさらに次のシーケンスを使用します。

```
nonl,intrflush(stdscr,FALSE),keypad(stdscr,TRUE);
```

curses のプログラムを実行する前に、端末のタブ位置を設定し、さらに初期化文字列群 (もし定義されていれば) を出力する必要があります。これを行うには、シェルの環境変数 TERM をエクスポートした後で tput init コマンドを実行してください。(詳しくは terminfo(4) の説明を参照してください)。

curses ライブラリは、「ウィンドウ」と呼ばれるデータ構造体を扱うことを可能にします。ウィンドウは、CRT 画面の全部分または一部分を表示している、二次元の文字配列と考えられます。デフォルトのウィンドウとして、端末画面と同一サイズの stdscr が提供されています。他のウィンドウは newwin(3CURSES) を使って作成できます。

WINDOW * と宣言された変数を使ってウィンドウを参照することができます。ウィンドウの操作は、セクション 3CURSES で説明されている curs_ で始まる名前を持つルーチンを使って行います。そのうち move(3CURSES) と addch(3CURSES) が最も基本的なルーチンです。その他にも、ユーザーがウィンドウを指定できるよう、w で始まる名前の汎用的な各種ルーチンが用意されています。w で始まらないルーチンは、stdscr に影響を与えます。

ウィンドウ操作用のルーチンを使用した後、ユーザーの CRT 画面を stdscr のように表示する旨を curses に指示する目的で、refresh(3CURSES) が呼び出されます。ウィンドウ内の文字の実際のタイプは chtype つまり文字プラス属性であり、各文字とともにその文字に関する情報が記録されています。

「パッド」と呼ばれる特殊なウィンドウも操作することができます。パッドとは、画面のサイズに依存しないウィンドウで、その内容は、すべてが一度に表示されるという必要はないものです。詳しくは curs_pad(3CURSES) の説明を参照してください。

画面上に文字を描くだけでなく、ビデオ属性やカラーも定義することができます。つまりアンダーラインなどの拡張機能や、リバース表示やカラーを端末がサポートしていれば、文字をそのようなモードで表示することも可能です。また線引き文字、すなわち線を描くための文字を出力するよう指定することもできます。入力時には、curses は矢印キーやファンクションキーを解釈して変換することもできます。これらのキーは、エスケープシーケンスを単一の値に転送するものです。ビデオ属性、線引きモード、および入力値は、A_REVERSE、ACS_HLINE、KEY_LEFT などの名前 (いずれも <curses.h> 中に定義されている) を使用します。

LINES、COLUMNS 環境変数が設定されている場合、およびプログラムがウィンドウ環境で実行中の場合、行とカラムに関する環境変数の情報が *terminfo* が読み込んだ情報に優先して用いられます。その結果、たとえば画面サイズが可変の AT&T 630 レイヤ内で実行中のプログラムは影響を受けます。

環境変数 TERMINFO が定義されているとき、curses を使うプログラムはまず標準ディレクトリを捜す前に、この TERMINFO で与えられたディレクトリをチェックします。たとえば TERM が att4424 に設定されていれば、コンパイルされた端末定義は中に見つかります。(この名前のうちの a は att4424 の先頭文字を流用したもので、ディレクトリが大きくなりすぎるのを防ぐための手段です。)

```
/usr/share/lib/terminfo/a/att4424
```

なお TERMINFO が $HOME/myterms に設定されていれば、curses は次の順序でチェックします。

```
$HOME/myterms/a/att4424,

/usr/share/lib/terminfo/a/att4424.
```

この機能は、テスト的に定義体を開発する場合や /usr/share/lib/terminfo に対する書き込み権を持っていない場合に使用すると便利です。

<curses.h> 中に定義されている整数変数 LINES と COLS には、initscr によって画面のサイズの値が代入されます。TRUE と FALSE の両定数の値はそれぞれ 1 と 0 です。

curses ルーチンは WINDOW * curscr 変数も定義します。この変数は、たとえばガーベッジを含んでいる画面をクリアして 再表示させるような低レベルの操作に用いられます。この curscr を使用できるのは少数のルーチンに限られています。

国際機能

補助文字セットの文字 1 個を保持するのに要するバイト数とカラム数はロケール固有の値であり (ロケールカテゴリは LC_CTYPE)、文字クラステーブル中に指定することができます。

編集に関しては文字レベルでの操作が適しています。画面のフォーマットに関しては、画面上で任意に文字を移動させることは好ましくありません。

文字の上書きを行うルーチン (たとえば addch) は画面レベルで動作します。既存の文字と上書きする文字とでカラム数が異なる場合、「オーファン (孤児) カラム」が発生する可能性があります。オーファンカラムはバックグラウンド文字で埋められます。

文字の挿入を行うルーチン (たとえば insch) は、文字レベルで (すなわち文字境界で) 動作します。指定された文字はカーソルが指す文字の直前に挿入されます。カーソルが現文字のどのカラムを指していても同じです。挿入に先立ち、カーソルは現文字の先頭カラムを指すよう調整されます。

文字削除用のルーチン (たとえば delch) も、やはり文字レベル (文字境界) で動作します。つまりどのカラムを指していても、カーソルが現在指している文字が削除されます。削除に先立ち、カーソルは現文字の先頭カラムを指すよう調整されます。

複数カラム文字を行の最終カラムに置くことはできません。そのような配置を試みると、そのカラムには代わりにバックグラウンド文字が置かれます。またこのような動作によりオーファンカラムが発生した場合、そのオーファンカラムはバックグラウンド文字で埋められます。

ウィンドウに対するオーバラップおよび上書き処理を実行すると、それに先立ってそのウィンドウの辺上にある文字が上書きされます。その際、オーファンカラムは (もしあれば) 文字の操作時と同様に処理されます。

カーソルはウィンドウ内のどの位置にも置くことが可能です。カーソルが複数カラム文字の 2 番目以降のカラムを指している時の挿入もしくは削除処理は、カーソルをその文字の先頭カラムに移動してから実行されます。

ルーチン名と引数名

多くの curses ライブラリルーチンは複数個のバージョンを持っています。w という接頭辞を持つルーチンはウィンドウ引数を必要とし、p という接頭辞を持つルーチンはパッド引数を必要とします。これらの接頭辞がないルーチンは、stdscr を使うのが一般的です。

mv という接頭辞を持つルーチンは、指定された動作を開始する前に移動を行いますが、その移動先を示す *x* と *y* の座標値を必要とします。mv ルーチンは他のルーチンを呼び出す前に move(3CURSES) を呼び出します。なお座標値 *y* は常に (ウィンドウの) 行位置を、座標値 *x* は常にカラム位置を表します。左上角の座標値は (1,1) ではなく (0,0) です。

接頭辞が mvw のルーチンは、ウィンドウ引数と *x* および *y* 座標値の両方を使用します。必ずウィンドウ引数の方が先に指定されます。

いずれの場合も、*win* は対象となるウィンドウを、*pad* は対象となるパッドを表します。*win* も *pad* も常にタイプ WINDOW へのポインタです。

オプションを設定するルーチンは、値として TRUE または FALSE を持つブール型のフラグ *bf* を必要とします。このフラグのタイプは常に bool です。変数 *ch* および *attrs* のタイプは常に chtype です。WINDOW、SCREEN、bool、chtype の各タイプは <curses.h> 内に定義されています。またタイプ TERMINAL は <term.h> 内に定義されています。その他の引数はいずれも整数です。

ルーチン名インデックス

以下にリストするのは全 curses ルーチンの名前と、それぞれが説明されているマニュアルページの名前です。

ルーチン名	マニュアル上のルーチン名
addch	curs_addch(3CURSES)
addchnstr	curs_addchstr(3CURSES)
addchstr	curs_addchstr(3CURSES)
addnstr	curs_addstr(3CURSES)
addnwstr	curs_addwstr(3CURSES)
addstr	curs_addstr(3CURSES)
addwch	curs_addwch(3CURSES)
addwchnstr	curs_addwchstr(3CURSES)
addwchstr	curs_addwchstr(3CURSES)
addwstr	curs_addwstr(3CURSES)
adjcurspos	curs_alecompat(3CURSES)
attroff	curs_attr(3CURSES)
attron	curs_attr(3CURSES)
attrset	curs_attr(3CURSES)
baudrate	curs_termattrs(3CURSES)
beep	curs_beep(3CURSES)
bkgd	curs_bkgd(3CURSES)
bkgdset	curs_bkgd(3CURSES)
border	curs_border(3CURSES)
box	curs_border(3CURSES)
can_change_color	curs_color(3CURSES)
cbreak	curs_inopts(3CURSES)
clear	curs_clear(3CURSES)
clearok	curs_outopts(3CURSES)
clrtobot	curs_clear(3CURSES)

clrtoeol	curs_clear(3CURSES)
color_content	curs_color(3CURSES)
copywin	curs_overlay(3CURSES)
curs_set	curs_kernel(3CURSES)
def_prog_mode	curs_kernel(3CURSES)
def_shell_mode	curs_kernel(3CURSES)
del_curterm	curs_terminfo(3CURSES)
delay_output	curs_util(3CURSES)
delch	curs_delch(3CURSES)
deleteln	curs_deleteln(3CURSES)
delscreen	curs_initscr(3CURSES)
delwin	curs_window(3CURSES)
derwin	curs_window(3CURSES)
doupdate	curs_refresh(3CURSES)
dupwin	curs_window(3CURSES)
echo	curs_inopts(3CURSES)
echochar	curs_addch(3CURSES)
echowchar	curs_addwch(3CURSES)
endwin	curs_initscr(3CURSES)
erase	curs_clear(3CURSES)
erasechar	curs_termattrs(3CURSES)
filter	curs_util(3CURSES)
flash	curs_beep(3CURSES)
flushinp	curs_util(3CURSES)
getbegyx	curs_getyx(3CURSES)
getch	curs_getch(3CURSES)
getmaxyx	curs_getyx(3CURSES)
getnwstr	curs_getwstr(3CURSES)
getparyx	curs_getyx(3CURSES)
getstr	curs_getstr(3CURSES)
getsyx	curs_kernel(3CURSES)

getwch	curs_getwch(3CURSES)
getwin	curs_util(3CURSES)
getwstr	curs_getwstr(3CURSES)
getyx	curs_getyx(3CURSES)
halfdelay	curs_inopts(3CURSES)
has_colors	curs_color(3CURSES)
has_ic	curs_termattrs(3CURSES)
has_il	curs_termattrs(3CURSES)
idcok	curs_outopts(3CURSES)
idlok	curs_outopts(3CURSES)
immedok	curs_outopts(3CURSES)
inch	curs_inch(3CURSES)
inchnstr	curs_inchstr(3CURSES)
inchstr	curs_inchstr(3CURSES)
init_color	curs_color(3CURSES)
init_pair	curs_color(3CURSES)
initscr	curs_initscr(3CURSES)
innstr	curs_instr(3CURSES)
innwstr	curs_inwstr(3CURSES)
insch	curs_insch(3CURSES)
insdelln	curs_deleteln(3CURSES)
insertln	curs_deleteln(3CURSES)
insnstr	curs_insstr(3CURSES)
insnwstr	curs_inswstr(3CURSES)
insstr	curs_insstr(3CURSES)
instr	curs_instr(3CURSES)
inswch	curs_inswch(3CURSES)
inswstr	curs_inswstr(3CURSES)
intrflush	curs_inopts(3CURSES)
inwch	curs_inwch(3CURSES)
inwchnstr	curs_inwchstr(3CURSES)

inwchstr	curs_inwchstr(3CURSES)
inwstr	curs_inwstr(3CURSES)
is_linetouched	curs_touch(3CURSES)
is_wintouched	curs_touch(3CURSES)
isendwin	curs_initscr(3CURSES)
keyname	curs_util(3CURSES)
keypad	curs_inopts(3CURSES)
killchar	curs_termattrs(3CURSES)
leaveok	curs_outopts(3CURSES)
longname	curs_termattrs(3CURSES)
meta	curs_inopts(3CURSES)
move	curs_move(3CURSES)
movenextch	curs_alecompat(3CURSES)
moveprevch	curs_alecompat(3CURSES)
mvaddch	curs_addch(3CURSES)
mvaddchnstr	curs_addchstr(3CURSES)
mvaddchstr	curs_addchstr(3CURSES)
mvaddnstr	curs_addstr(3CURSES)
mvaddnwstr	curs_addwstr(3CURSES)
mvaddstr	curs_addstr(3CURSES)
mvaddwch	curs_addwch(3CURSES)
mvaddwchnstr	curs_addwchstr(3CURSES)
mvaddwchstr	curs_addwchstr(3CURSES)
mvaddwstr	curs_addwstr(3CURSES)
mvcur	curs_terminfo(3CURSES)
mvdelch	curs_delch(3CURSES)
mvderwin	curs_window(3CURSES)
mvgetch	curs_getch(3CURSES)
mvgetnwstr	curs_getwstr(3CURSES)
mvgetstr	curs_getstr(3CURSES)
mvgetwch	curs_getwch(3CURSES)

mvgetwstr	curs_getwstr(3CURSES)
mvinch	curs_inch(3CURSES)
mvinchnstr	curs_inchstr(3CURSES)
mvinchstr	curs_inchstr(3CURSES)
mvinnstr	curs_instr(3CURSES)
mvinnwstr	curs_inwstr(3CURSES)
mvinsch	curs_insch(3CURSES)
mvinsnstr	curs_insstr(3CURSES)
mvinsnwstr	curs_inswstr(3CURSES)
mvinsstr	curs_insstr(3CURSES)
mvinstr	curs_instr(3CURSES)
mvinswch	curs_inswch(3CURSES)
mvinswstr	curs_inswstr(3CURSES)
mvinwch	curs_inwch(3CURSES)
mvinwchnstr	curs_inwchstr(3CURSES)
mvinwchstr	curs_inwchstr(3CURSES)
mvinwstr	curs_inwstr(3CURSES)
mvprintw	curs_printw(3CURSES)
mvscanw	curs_scanw(3CURSES)
mvwaddch	curs_addch(3CURSES)
mvwaddchnstr	curs_addchstr(3CURSES)
mvwaddchstr	curs_addchstr(3CURSES)
mvwaddnstr	curs_addstr(3CURSES)
mvwaddnwstr	curs_addwstr(3CURSES)
mvwaddstr	curs_addstr(3CURSES)
mvwaddwch	curs_addwch(3CURSES)
mvwaddwchnstr	curs_addwchstr(3CURSES)
mvwaddwchstr	curs_addwchstr(3CURSES)
mvwaddwstr	curs_addwstr(3CURSES)
mvwdelch	curs_delch(3CURSES)
mvwgetch	curs_getch(3CURSES)

mvwgetnwstr	curs_getwstr(3CURSES)
mvwgetstr	curs_getstr(3CURSES)
mvwgetwch	curs_getwch(3CURSES)
mvwgetwstr	curs_getwstr(3CURSES)
mvwin	curs_window(3CURSES)
mvwinch	curs_inch(3CURSES)
mvwinchnstr	curs_inchstr(3CURSES)
mvwinchstr	curs_inchstr(3CURSES)
mvwinnstr	curs_instr(3CURSES)
mvwinnwstr	curs_inwstr(3CURSES)
mvwinsch	curs_insch(3CURSES)
mvwinsnstr	curs_insstr(3CURSES)
mvwinsstr	curs_insstr(3CURSES)
mvwinstr	curs_instr(3CURSES)
mvwinswch	curs_inswch(3CURSES)
mvwinswstr	curs_inswstr(3CURSES)
mvwinwch	curs_inwch(3CURSES)
mvwinwchnstr	curs_inwchstr(3CURSES)
mvwinwchstr	curs_inwchstr(3CURSES)
mvwinwstr	curs_inwstr(3CURSES)
mvwprintw	curs_printw(3CURSES)
mvwscanw	curs_scanw(3CURSES)
napms	curs_kernel(3CURSES)
newpad	curs_pad(3CURSES)
newterm	curs_initscr(3CURSES)
newwin	curs_window(3CURSES)
nl	curs_outopts(3CURSES)
nocbreak	curs_inopts(3CURSES)
nodelay	curs_inopts(3CURSES)
noecho	curs_inopts(3CURSES)
nonl	curs_outopts(3CURSES)

noqiflush	curs_inopts(3CURSES)
noraw	curs_inopts(3CURSES)
notimeout	curs_inopts(3CURSES)
overlay	curs_overlay(3CURSES)
overwrite	curs_overlay(3CURSES)
pair_content	curs_color(3CURSES)
pechochar	curs_pad(3CURSES)
pechowchar	curs_pad(3CURSES)
pnoutrefresh	curs_pad(3CURSES)
prefresh	curs_pad(3CURSES)
printw	curs_printw(3CURSES)
putp	curs_terminfo(3CURSES)
putwin	curs_util(3CURSES)
qiflush	curs_inopts(3CURSES)
raw	curs_inopts(3CURSES)
redrawwin	curs_refresh(3CURSES)
refresh	curs_refresh(3CURSES)
reset_prog_mode	curs_kernel(3CURSES)
reset_shell_mode	curs_kernel(3CURSES)
resetty	curs_kernel(3CURSES)
restartterm	curs_terminfo(3CURSES)
ripoffline	curs_kernel(3CURSES)
savetty	curs_kernel(3CURSES)
scanw	curs_scanw(3CURSES)
scr_dump	curs_scr_dump(3CURSES)
scr_init	curs_scr_dump(3CURSES)
scr_restore	curs_scr_dump(3CURSES)
scr_set	curs_scr_dump(3CURSES)
scroll	curs_scroll(3CURSES)
scrollok	curs_outopts(3CURSES)
set_curterm	curs_terminfo(3CURSES)

set_term	curs_initscr(3CURSES)
setscrreg	curs_outopts(3CURSES)
setsyx	curs_kernel(3CURSES)
setterm	curs_terminfo(3CURSES)
setupterm	curs_terminfo(3CURSES)
slk_attroff	curs_slk(3CURSES)
slk_attron	curs_slk(3CURSES)
slk_attrset	curs_slk(3CURSES)
slk_clear	curs_slk(3CURSES)
slk_init	curs_slk(3CURSES)
slk_label	curs_slk(3CURSES)
slk_noutrefresh	curs_slk(3CURSES)
slk_refresh	curs_slk(3CURSES)
slk_restore	curs_slk(3CURSES)
slk_set	curs_slk(3CURSES)
slk_touch	curs_slk(3CURSES)
srcl	curs_scroll(3CURSES)
standend	curs_attr(3CURSES)
standout	curs_attr(3CURSES)
start_color	curs_color(3CURSES)
subpad	curs_pad(3CURSES)
subwin	curs_window(3CURSES)
syncok	curs_window(3CURSES)
termattrs	curs_termattrs(3CURSES)
termname	curs_termattrs(3CURSES)
tgetent	curs_termcap(3CURSES)
tgetflag	curs_termcap(3CURSES)
tgetnum	curs_termcap(3CURSES)
tgetstr	curs_termcap(3CURSES)
tgoto	curs_termcap(3CURSES)
tigetflag	curs_terminfo(3CURSES)

tigetnum	curs_terminfo(3CURSES)
tigetstr	curs_terminfo(3CURSES)
timeout	curs_inopts(3CURSES)
touchline	curs_touch(3CURSES)
touchwin	curs_touch(3CURSES)
tparm	curs_terminfo(3CURSES)
tputs	curs_terminfo(3CURSES)
typeahead	curs_inopts(3CURSES)
unctrl	curs_util(3CURSES)
ungetch	curs_getch(3CURSES)
ungetwch	curs_getwch(3CURSES)
untouchwin	curs_touch(3CURSES)
use_env	curs_util(3CURSES)
vidattr	curs_terminfo(3CURSES)
vidputs	curs_terminfo(3CURSES)
vwprintw	curs_printw(3CURSES)
vwscanw	curs_scanw(3CURSES)
waddch	curs_addch(3CURSES)
waddchnstr	curs_addchstr(3CURSES)
waddchstr	curs_addchstr(3CURSES)
waddnstr	curs_addstr(3CURSES)
waddnwstr	curs_addwstr(3CURSES)
waddstr	curs_addstr(3CURSES)
waddwch	curs_addwch(3CURSES)
waddwchnstr	curs_addwchstr(3CURSES)
waddwchstr	curs_addwchstr(3CURSES)
waddwstr	curs_addwstr(3CURSES)
wadjcurspos	curs_alecompat(3CURSES)
wattroff	curs_attr(3CURSES)
wattron	curs_attr(3CURSES)
wattrset	curs_attr(3CURSES)

wbkgd	curs_bkgd(3CURSES)
wbkgdset	curs_bkgd(3CURSES)
wborder	curs_border(3CURSES)
wclear	curs_clear(3CURSES)
wclrtobot	curs_clear(3CURSES)
wclrtoeol	curs_clear(3CURSES)
wcursyncup	curs_window(3CURSES)
wdelch	curs_delch(3CURSES)
wdeleteln	curs_deleteln(3CURSES)
wechochar	curs_addch(3CURSES)
wechowchar	curs_addwch(3CURSES)
werase	curs_clear(3CURSES)
wgetch	curs_getch(3CURSES)
wgetnstr	curs_getstr(3CURSES)
wgetnwstr	curs_getwstr(3CURSES)
wgetstr	curs_getstr(3CURSES)
wgetwch	curs_getwch(3CURSES)
wgetwstr	curs_getwstr(3CURSES)
whline	curs_border(3CURSES)
winch	curs_inch(3CURSES)
winchnstr	curs_inchstr(3CURSES)
winchstr	curs_inchstr(3CURSES)
winnstr	curs_instr(3CURSES)
winnwstr	curs_inwstr(3CURSES)
winsch	curs_insch(3CURSES)
winsdelln	curs_deleteln(3CURSES)
winsertln	curs_deleteln(3CURSES)
winsnstr	curs_insstr(3CURSES)
winsnwstr	curs_inswstr(3CURSES)
winsstr	curs_insstr(3CURSES)
winstr	curs_instr(3CURSES)

winswch	curs_inswch(3CURSES)
winswstr	curs_inswstr(3CURSES)
winwch	curs_inwch(3CURSES)
winwchnstr	curs_inwchstr(3CURSES)
winwchstr	curs_inwchstr(3CURSES)
winwstr	curs_inwstr(3CURSES)
wmove	curs_move(3CURSES)
wmovenextch	curs_alecompat(3CURSES)
wmoveprevch	curs_alecompat(3CURSES)
wnoutrefresh	curs_refresh(3CURSES)
wprintw	curs_printw(3CURSES)
wredrawln	curs_refresh(3CURSES)
wrefresh	curs_refresh(3CURSES)
wscanw	curs_scanw(3CURSES)
wscrl	curs_scroll(3CURSES)
wsetscrreg	curs_outopts(3CURSES)
wstandend	curs_attr(3CURSES)
wstandout	curs_attr(3CURSES)
wsyncdown	curs_window(3CURSES)
wsyncup	curs_window(3CURSES)
wtimeout	curs_inopts(3CURSES)
wtouchln	curs_touch(3CURSES)
wvline	curs_border(3CURSES)

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

setscrreg()、wsetscrreg()、getyx()、getbegyx()、および getmaxyx() を除くすべてのルーチンは、w バージョンの値を返します。setscrreg()、wsetscrreg()、getyx()、getbegyx()、および getmaxyx() が返す値は未定義です。したがって、これらを代入文の右辺に用いるべきではありません。

ポインタを返すルーチンは、エラーが発生した場合には NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 curses(3XCURSES), libcurses(3LIB), libcurses(3XCURSES), terminfo(4), attributes(5)

注意事項 ヘッダー <curses.h> は自動的に <stdio.h> および <unctrl.h> ヘッダーを含みます。

名前

curs_addwch, addwch, waddwch, mvaddwch, mvwaddwch, echowchar, wechowchar – wchar_t 文字を属性とともに curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwch(chtype wch);

int waddwch(WINDOW *win, chtype wch);

int mvaddwch(int y, int x, chtype wch);

int mvwaddwch(WINDOW *win, int y, int x, chtype wch);

int echowchar(chtype wch);

int wechowchar(WINDOW *win, chtype wch);
```

機能説明

addwch()、waddwch()、mvaddwch()、mvwaddwch() の各ルーチンは、wchar_t を保ちながら *wch* 文字をウィンドウ中の現カーソル位置に書き出し、ウィンドウカーソルの位置を進めます。この機能は C 複数バイトライブラリ内にある putwchar(3C) の機能に似ています。右側のマージンでは自動的に復帰改行 (newline) が行われます。scrollok が有効であれば、スクロール領域が上に 1 行スクロールされます。

wch がタブ、復帰改行、バックスペースのいずれかの場合、それに従ってウィンドウ内でカーソルが移動します。なお復帰改行の場合には、カーソル移動に先だって clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wch* が他の制御文字の場合、^*X* 形式で描かれます。制御文字を追加した後で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

ビデオ属性は、論理和をパラメータ中にとることにより wchar_t 文字と組み合わせることができます。その結果、属性の設定も行われます。これが意味することは、inwch() と addwch() を使ってテキスト (属性も含む) をある場所から他の場所へコピーすることが可能であるということです。standout(3CURSES) の、定義済みビデオ属性定数の項を参照してください。

機能的に見ると、echowchar() ルーチンは addwch() と refresh(3CURSES) を連続して呼び出すことと同等で、wechowchar() ルーチンは waddwch() と wrefresh(3CURSES) を連続して呼び出すことと同等です。両ルーチンは 1 文字だけしか出力されないことが判っていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。

線グラフィック

addwch() ファミリのルーチンとともに以下の変数を使用して、線を描画する文字を画面に書き出すことができます。端末に変数が定義されているとき、A_ALTCHARSET ビットがオンになります (curs_attr(3CURSES) の項を参照)。変数が定義されていなければ、以下に述べるデフォルト文字が変数に与えられます。なお各変数の名前は VT100 命名規約に準拠しています。

名前	デフォルト	グリフ記述
ACS_ULCORNER	+	左上角
ACS_LLCORNER	+	左下角
ACS_URCORNER	+	右上角
ACS_LRCORNER	+	右下角
ACS_RTEE	+	右端
ACS_LTEE	+	左端
ACS_BTEE	+	下端
ACS_TTEE	+	上端
ACS_HLINE	–	横線
ACS_VLINE	|	縦線
ACS_PLUS	+	正符号
ACS_S1	-	スキャンライン 1
ACS_S9	_	スキャンライン 9
ACS_DIAMOND	+	ダイヤモンド
ACS_CKBOARD	:	市松模様 (点描)
ACS_DEGREE	'	度数記号
ACS_PLMINUS	#	プラス／マイナス
ACS_BULLET	o	丸
ACS_LARROW	<	左向きの矢印
ACS_RARROW	>	右向きの矢印
ACS_DARROW	v	下向きの矢印
ACS_UARROW	^	上向きの矢印
ACS_BOARD	#	正方形のボード
ACS_LANTERN	#	ランタン記号
ACS_BLOCK	#	正方形のブロック

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 putwchar(3C), clrtoeol(3CURSES), curses(3CURSES), curs_attr(3CURSES), curs_inwch(3CURSES), curs_outopts(3CURSES), refresh(3CURSES), standout(3CURSES), winwch(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

addwch()、mvaddwch()、mvwaddwch()、echowchar() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明

上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwchnstr() 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_alecompat, movenextch, wmovenextch, moveprevch, wmoveprevch, adjcurspos, wadjcurspos – カーソルを文字単位に移動するために ALE curses ライブラリに追加された関数

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int movenextch(void);

int wmovenextch(WINDOW *win);

int moveprevch(void);

int wmoveprevch(WINDOW *win);

int adjcurspos(void);

int wadjcurspos(WINDOW *win);
```

機能説明

上記 `movenextch()` および `wmovenextch()` ルーチンは、カーソルを次の (右方の) 文字に移動します。次の文字が複数カラム文字であれば、新カーソル位置はその文字の最初の (左端の) カラムとなります。現カーソル位置が複数カラム文字の左端カラムであっても、新しい位置は次の文字上に移動します。なお現カーソル位置が複数カラム文字上の場合、単純なカーソル移動 (++x) を実行しても次の文字に移動するとは保証できません。新しいカーソル位置は `getyx`(3CURSES) を実行することにより得られます。

`moveprevc()` および `wmoveprevch()` ルーチンは、上記 `movenextch()` と `wmovenextch()` とは反対方向に、つまり 1 つ前の文字の左端カラムにカーソルを移動します。

`adjcurspos()` および `wadjcurspos()` ルーチンは、現在カーソルが置かれている複数カラム文字の左端のカラムへカーソルを移動します。現在の位置がすでに左端カラムである場合、および現在置かれている文字がシングルカラム文字の場合、これらのルーチンは意味を持たず、カーソル位置は変わりません。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性

次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

`curses`(3CURSES), `getyx`(3CURSES), `attributes`(5)

注意事項

ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`movenextch()`、`moveprevch()`、`adjcurspos()` の各ルーチンはマクロにすることも可能です。

名前

curs_getwch, getwch, wgetwch, mvgetwch, mvwgetwch, ungetwch – curses 端末キーボードから wchar_t 文字を得る (または戻す)

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwch(void);

int wgetwch(WINDOW *win);

int mvgetwch(int y, int x);

int mvwgetwch(WINDOW *win, int y, int x);

int ungetwch(int wch);
```

機能説明

getwch()、wgetwch()、mvgetwch()、mvwgetwch() の各ルーチンは、ウィンドウに対応づけられた端末から EUC 文字 1 文字を読み込み、それを wchar_t 文字に転送し、wchar_t 文字を返します。遅延無しモードでは、入力待ちのものがなければ ERR 値が返されます。遅延モードでは、システムからテキストが渡されるまでプログラムは待ち状態に置かれます。待つタイミングは cbreak の設定により異なり、1 つの文字の後 (cbreak モード) または最初の改行の後 (nocbreak モード) となります。また半遅延モードでは、何らかの文字が入力されるまであるいは指定されたタイムアウト値に達するまで待ちます。noecho が設定されていなければ、読み込まれた文字は指定されたウィンドウ中にエコーされます。

ウィンドウがパッドでなく、最後に wrefresh(3CURSES) が呼び出されてからウィンドウの移動または変更が行われていれば、次の文字を読み込む前に wrefresh が呼び出されます。

keypad が TRUE の場合、ファンクションキーが押されると実際の文字の代わりにファンクションキーのトークンが返されます。使用可能なファンクションキーは KEY_ で始まる名前で、0401 で始まる整数とともに <curses.h> 中に定義されています。ファンクションキーの先頭文字でありうる文字 (たとえばエスケープ文字) を受け取ると、curses(3CURSES) はタイマをセットします。指定された時間内にそのシーケンスの残りの文字が到着しなければ、受け取った 1 文字だけを渡します。時間内に残りの文字を受け取れば、ファンクションキーの値を返します。このため、エスケープ文字を押したのにそれがプログラムに渡されるまでに時間がかかった、 というケースを多くの端末が経験することになります。

ungetwch() ルーチンは、次に wgetwch() を呼び出したときに返される入力キュー中に *wch* 文字を戻します。

ファンクションキー

keypad が有効であれば、以下の表にリストされているファンクションキー (いずれも <curses.h> 中に定義) が getwch() により返されます。なお端末によっては、これらすべてのファンクションキーがサポートされるとは限りません。サポートされないケースは、そのキーが押されたときにその端末転送するコードが一意ではない場合、およびそのキーの定義が terminfo(4) データベース中に存在しない場合です。

名前	キー名
KEY_BREAK	Break キー
KEY_DOWN	4 つの矢印キー ...
KEY_UP	
KEY_LEFT	
KEY_RIGHT	
KEY_HOME	Home キー (左上向き矢印)
KEY_BACKSPACE	バックスペース
KEY_F0	ファンクションキー (64 個分の空白が予約されている)
KEY_F(*n*)	For $0 \leq n \leq 63$
KEY_DL	行削除
KEY_IL	行挿入
KEY_DC	文字削除
KEY_IC	文字挿入、または挿入モードに入る
KEY_EIC	挿入モードを抜ける
KEY_CLEAR	画面クリア
KEY_EOS	画面の終わりまでクリア
KEY_EOL	行の終わりまでクリア
KEY_SF	上方へ 1 行スクロール
KEY_SR	下方へ 1 行スクロール (リバース)
KEY_NPAGE	次ページ
KEY_PPAGE	前ページ
KEY_STAB	タブを設定
KEY_CTAB	タブをクリア
KEY_CATAB	全タブをクリア
KEY_ENTER	入力または送信
KEY_SRESET	ソフト (部分的) リセット
KEY_RESET	リセットまたはハードリセット
KEY_PRINT	印刷またはコピー

名前	キー名
KEY_LL	下部のホーム位置 (左下) へ。 キーパッドの割り当ては次のとおり。 A1 up A3 left B2 right C1 down C3
KEY_A1	キーパッドの左上
KEY_A3	キーパッドの右上
KEY_B2	キーパッドの中央
KEY_C1	キーパッドの左下
KEY_C3	キーパッドの右下
KEY_BTAB	バックタブキー
KEY_BEG	開始 (Beginning) キー
KEY_CANCEL	キャンセルキー
KEY_CLOSE	クローズキー
KEY_COMMAND	コマンド (cmd) キー
KEY_COPY	コピーキー
KEY_CREATE	作成キー
KEY_END	終了キー
KEY_EXIT	Exit キー
KEY_FIND	検索キー
KEY_HELP	ヘルプキー
KEY_MARK	マークキー
KEY_MESSAGE	メッセージキー
KEY_MOVE	移動キー
KEY_NEXT	次オブジェクトキー
KEY_OPEN	オープンキー
KEY_OPTIONS	オプションキー
KEY_PREVIOUS	前オブジェクトキー
KEY_REDO	再実行キー
KEY_REFERENCE	参照キー
KEY_REFRESH	リフレッシュキー

名前	キー名
KEY_REPLACE	置換キー
KEY_RESTART	再スタートキー
KEY_RESUME	再開キー
KEY_SAVE	セーブキー
KEY_SBEG	シフト付き開始キー
KEY_SCANCEL	シフト付きキャンセルキー
KEY_SCOMMAND	シフト付きコマンドキー
KEY_SCOPY	シフト付きコピーキー
KEY_SCREATE	シフト付き作成キー
KEY_SDC	シフト付き文字削除キー
KEY_SDL	シフト付き行削除キー
KEY_SELECT	選択キー
KEY_SEND	シフト付き送信キー
KEY_SEOL	シフト付き行クリアキー
KEY_SEXIT	シフト付き Exit キー
KEY_SFIND	シフト付き検索キー
KEY_SHELP	シフト付きヘルプキー
KEY_SHOME	シフト付き Home キー
KEY_SIC	シフト付き入力キー
KEY_SLEFT	シフト付き左向き矢印キー
KEY_SMESSAGE	シフト付きメッセージキー
KEY_SMOVE	シフト付き移動キー
KEY_SNEXT	シフト付き次オブジェクトキー
KEY_SOPTIONS	シフト付きオプションキー
KEY_SPREVIOUS	シフト付き前オブジェクトキー
KEY_SPRINT	シフト付き印刷キー
KEY_SREDO	シフト付き再実行キー
KEY_SREPLACE	シフト付き置換キー
KEY_SRIGHT	シフト付き右向き矢印キー

名前	キー名
KEY_SRSUME	シフト付き再開キー
KEY_SSAVE	シフト付きセーブキー
KEY_SSUSPEND	シフト付き中断キー
KEY_SUNDO	シフト付き取消キー
KEY_SUSPEND	中断キー
KEY_UNDO	取消キー

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目　curses(3CURSES), curs_inopts(3CURSES), curs_move(3CURSES), wrefresh(3CURSES), terminfo(4), attributes(5)

注意事項　ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

単一文字関数に対してエスケープ文字を使うことはできません。

getwch()、wgetwch()、mvgetwch()、mvwgetwch() のいずれかの関数を用いる際、nocbreak モードと echo モードを同時に使用することはできません。個々の文字が入力されたときの tty ドライバの状態によっては、プログラムは予測したものとは異なる結果を生成する場合があります。

getwch()、mvgetwch()、mvwgetwch() の各ルーチンはマクロにすることも可能です。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inswch, inswch, winswch, mvinswch, mvwinswch – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswch(chtype wch);

int winswch(WINDOW *win, chtype wch);

int mvinswch(int y, int x, chtype wch);

int mvwinswch(WINDOW *win, int y, int x, chtype wch);
```

機能説明 | これらのルーチンは、wchar_t 文字を含んでいる *wch* を、現カーソル位置の文字の直前に挿入します。これによりカーソルの右側にある文字はそれぞれ 1 文字分ずつ右に移動し、その結果、行の右端にある文字が失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用する ということを意味するものではありません。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

inswch()、mvinswch()、mvwinswch() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式 |

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^*X* 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inwch, inwch, winwch, mvinwch, mvwinwch – curses ウィンドウから wchar_t 文字とその属性を得る

形式 |
```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

chtype inwch(void);

chtype winwch(WINDOW *win);

chtype mvinwch(int y, int x);

chtype mvwinwch(WINDOW *win, int y, int x);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置にある、chtype タイプの wchar_t 文字を返します。その位置に属性が設定されていれば、それらの属性値の論理和も返されます。<curses.h> 中に定義されている定数を & (論理積) 演算子とともに用いて、文字だけもしくは属性だけを抽出することも可能です。

属性 | winwch() が返す値と以下に示すビットマスクとの論理積をとることもできます。

A_WCHARTEXT	文字抽出用のビットマスク
A_WATTRIBUTES	属性抽出用のビットマスク

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

全ルーチンともマクロにすることが可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式 |
```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の `wchar_t` 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の `wchar_t` 文字を返します。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性 | 次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | `curses`(3CURSES), `attributes`(5)

注意事項 | ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`winnwstr()` 以外のルーチンはマクロにすることも可能です。

名前

curs_pad, newpad, subpad, prefresh, pnoutrefresh, pechochar, pechowchar – curses パッドの作成と表示

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);

WINDOW *subpad(WINDOW *orig, int nlines, int ncols, int begin_y, int
     begin_x);

int prefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow, int
     smincol, int smaxrow, int smaxcol);

int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow,
     int smincol, int smaxrow, int smaxcol);

int pechochar(WINDOW *pad, chtype ch);

int pechowchar(WINDOW *pad, chtype wch);
```

機能説明

newpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるパッドデータ構造体を作成し、その構造体へのポインタを返します。パッドはウィンドウに似ていますが、ウィンドウとは異なり、画面のサイズにより制限されたり 画面上の特別な部分に対応したりしているとは限りません。パッドは大きなウィンドウが必要なときに使うことができ、同時にウィンドウの一部だけが画面上に表示されます。パッドに対しては (入力のスクロールやエコーなどからの) 自動 リフレッシュ処理は行われません。引数に *pad* を指定して wrefresh(3CURSES) を呼び出すことはできませんので、リフレッシュを行いたければ prefresh() または pnoutrefresh() ルーチンを呼び出してください。これらのルーチンに対しては、表示対象とするパッドの部分および 表示に用いる画面の位置を指定するパラメータを与える必要があります。

subpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるサブウィンドウをパッド内に作成し、そのサブウィンドウへのポインタを返します。 画面上の座標値を用いる subwin(3CURSES) とは異なり、ウィンドウはパッド上の (*begin_x, begin_y*) に作成されます。また新ウィンドウは元のウィンドウ *orig* 内に置かれるので、一方のウィンドウで行った修正は他方のウィンドウにも適用されます。本ルーチン使用時には、prefresh() を呼び出す前に *orig* 上で touchwin(3CURSES) または touchline(3CURSES) を呼び出す必要が生じる場合がよくあります。

prefresh() と pnoutrefresh() の両ルーチンは、wrefresh(3CURSES) と wnoutrefresh(3CURSES) の両ルーチンと同等機能ですが、ウィンドウではなくパッドを処理対象とする点が異なります。これらを呼び出す際、パッドおよび画面のどの部分を対象とするかを指定するパラメータが必要となります。*pminrow* と *pmincol* は、パッド中に表示する四角形の左上角の位置を指定します。*sminrow*、*smincol*、*smaxrow*、*smaxcol* の各引数は、画面に表示する四角形の各辺の位置を指定します。四角形の大きさは常に同じなので、パッド中の四角形右下角の位置は画面座標値から自動的に計算されます。この 2 つの四角形は、ともにそれぞれの構造体の中に完全に納まっていなければなりません。*pminrow*、*pmincol*、*sminrow*、*smincol* の各引数が負の値のときはゼロと見なされます。

pechochar() ルーチンは、機能的には addch(3CURSES) と refresh(3CURSES)、waddch(3CURSES) と wrefresh(3CURSES)、または waddch(3CURSES) と prefresh() を、それぞれ連続して呼び出すことと同等です。本ルーチンは 1 文字だけしか出力されないことがわかっていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較して パフォーマンスの大幅な向上が期待できます。pechochar() の場合、画面上のパッドの最新の位置が prefresh() への引数として再利用されます。

戻り値

整数値を返すルーチンは、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

ポインタを返すルーチンは、エラーが発生すれば NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

addch(3CURSES), curses(3CURSES), refresh(3CURSES), subwin(3CURSES), touchline(3CURSES), touchwin(3CURSES), waddch(3CURSES), wnoutrefresh(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項

ヘッダー <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーを含みます。

pechochar() はマクロにすることも可能です。

名前

curs_addwch, addwch, waddwch, mvaddwch, mvwaddwch, echowchar, wechowchar – wchar_t 文字を属性とともに curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwch(chtype wch);

int waddwch(WINDOW *win, chtype wch);

int mvaddwch(int y, int x, chtype wch);

int mvwaddwch(WINDOW *win, int y, int x, chtype wch);

int echowchar(chtype wch);

int wechowchar(WINDOW *win, chtype wch);
```

機能説明

addwch()、waddwch()、mvaddwch()、mvwaddwch() の各ルーチンは、wchar_t を保ちながら *wch* 文字をウィンドウ中の現カーソル位置に書き出し、ウィンドウカーソルの位置を進めます。この機能は C 複数バイトライブラリ内にある putwchar(3C) の機能に似ています。右側のマージンでは自動的に復帰改行 (newline) が行われます。scrollok が有効であれば、スクロール領域が上に 1 行スクロールされます。

wch がタブ、復帰改行、バックスペースのいずれかの場合、それに従ってウィンドウ内でカーソルが移動します。なお復帰改行の場合には、カーソル移動に先だって clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wch* が他の制御文字の場合、^*X* 形式で描かれます。制御文字を追加した後で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

ビデオ属性は、論理和をパラメータ中にとることにより wchar_t 文字と組み合わせることができます。その結果、属性の設定も行われます。これが意味することは、inwch() と addwch() を使ってテキスト (属性も含む) をある場所から他の場所へコピーすることが可能であるということです。standout(3CURSES) の、定義済みビデオ属性定数の項を参照してください。

機能的に見ると、echowchar() ルーチンは addwch() と refresh(3CURSES) を連続して呼び出すことと同等で、wechowchar() ルーチンは waddwch() と wrefresh(3CURSES) を連続して呼び出すことと同等です。両ルーチンは 1 文字だけしか出力されないことが判っていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。

線グラフィック

addwch() ファミリのルーチンとともに以下の変数を使用して、線を描画する文字を画面に書き出すことができます。端末に変数が定義されているとき、A_ALTCHARSET ビットがオンになります (curs_attr(3CURSES) の項を参照)。変数が定義されていなければ、以下に述べるデフォルト文字が変数に与えられます。なお各変数の名前は VT100 命名規約に準拠しています。

名前	デフォルト	グリフ記述
ACS_ULCORNER	+	左上角
ACS_LLCORNER	+	左下角
ACS_URCORNER	+	右上角
ACS_LRCORNER	+	右下角
ACS_RTEE	+	右端
ACS_LTEE	+	左端
ACS_BTEE	+	下端
ACS_TTEE	+	上端
ACS_HLINE	–	横線
ACS_VLINE	|	縦線
ACS_PLUS	+	正符号
ACS_S1	-	スキャンライン 1
ACS_S9	_	スキャンライン 9
ACS_DIAMOND	+	ダイヤモンド
ACS_CKBOARD	:	市松模様 (点描)
ACS_DEGREE	'	度数記号
ACS_PLMINUS	#	プラス／マイナス
ACS_BULLET	o	丸
ACS_LARROW	<	左向きの矢印
ACS_RARROW	>	右向きの矢印
ACS_DARROW	v	下向きの矢印
ACS_UARROW	^	上向きの矢印
ACS_BOARD	#	正方形のボード
ACS_LANTERN	#	ランタン記号
ACS_BLOCK	#	正方形のブロック

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 putwchar(3C), clrtoeol(3CURSES), curses(3CURSES), curs_attr(3CURSES), curs_inwch(3CURSES), curs_outopts(3CURSES), refresh(3CURSES), standout(3CURSES), winwch(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

addwch()、mvaddwch()、mvwaddwch()、echowchar() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_getwch, getwch, wgetwch, mvgetwch, mvwgetwch, ungetwch – curses 端末キーボードから wchar_t 文字を得る (または戻す)

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwch(void);

int wgetwch(WINDOW *win);

int mvgetwch(int y, int x);

int mvwgetwch(WINDOW *win, int y, int x);

int ungetwch(int wch);
```

機能説明

getwch()、wgetwch()、mvgetwch()、mvwgetwch() の各ルーチンは、ウィンドウに対応づけられた端末から EUC 文字 1 文字を読み込み、それを wchar_t 文字に転送し、wchar_t 文字を返します。遅延無しモードでは、入力待ちのものがなければ ERR 値が返されます。遅延モードでは、システムからテキストが渡されるまでプログラムは待ち状態に置かれます。待つタイミングは cbreak の設定により異なり、1 つの文字の後 (cbreak モード) または最初の改行の後 (nocbreak モード) となります。また半遅延モードでは、何らかの文字が入力されるまであるいは指定されたタイムアウト値に達するまで待ちます。noecho が設定されていなければ、読み込まれた文字は指定されたウィンドウ中にエコーされます。

ウィンドウがパッドでなく、最後に wrefresh(3CURSES) が呼び出されてからウィンドウの移動または変更が行われていれば、次の文字を読み込む前に wrefresh が呼び出されます。

keypad が TRUE の場合、ファンクションキーが押されると実際の文字の代わりにファンクションキーのトークンが返されます。使用可能なファンクションキーは KEY_ で始まる名前で、0401 で始まる整数とともに <curses.h> 中に定義されています。ファンクションキーの先頭文字でありうる文字 (たとえばエスケープ文字) を受け取ると、curses(3CURSES) はタイマをセットします。指定された時間内にそのシーケンスの残りの文字が到着しなければ、受け取った 1 文字だけを渡します。時間内に残りの文字を受け取れば、ファンクションキーの値を返します。このため、エスケープ文字を押したのにそれがプログラムに渡されるまでに時間がかかった、 というケースを多くの端末が経験することになります。

ungetwch() ルーチンは、次に wgetwch() を呼び出したときに返される入力キュー中に *wch* 文字を戻します。

ファンクションキー

keypad が有効であれば、以下の表にリストされているファンクションキー (いずれも <curses.h> 中に定義) が getwch() により返されます。なお端末によっては、これらすべてのファンクションキーがサポートされるとは限りません。サポートされないケースは、そのキーが押されたときにその端末転送するコードが一意ではない場合、およびそのキーの定義が terminfo(4) データベース中に存在しない場合です。

名前	キー名
KEY_BREAK	Break キー
KEY_DOWN	4 つの矢印キー ...
KEY_UP	
KEY_LEFT	
KEY_RIGHT	
KEY_HOME	Home キー (左上向き矢印)
KEY_BACKSPACE	バックスペース
KEY_F0	ファンクションキー (64 個分の空白が予約されている)
KEY_F(*n*)	For $0 \leq n \leq 63$
KEY_DL	行削除
KEY_IL	行挿入
KEY_DC	文字削除
KEY_IC	文字挿入、または挿入モードに入る
KEY_EIC	挿入モードを抜ける
KEY_CLEAR	画面クリア
KEY_EOS	画面の終わりまでクリア
KEY_EOL	行の終わりまでクリア
KEY_SF	上方へ 1 行スクロール
KEY_SR	下方へ 1 行スクロール (リバース)
KEY_NPAGE	次ページ
KEY_PPAGE	前ページ
KEY_STAB	タブを設定
KEY_CTAB	タブをクリア
KEY_CATAB	全タブをクリア
KEY_ENTER	入力または送信
KEY_SRESET	ソフト (部分的) リセット
KEY_RESET	リセットまたはハードリセット
KEY_PRINT	印刷またはコピー

名前	キー名
KEY_LL	下部のホーム位置 (左下) へ。 キーパッドの割り当ては次のとおり。 A1 up A3 left B2 right C1 down C3
KEY_A1	キーパッドの左上
KEY_A3	キーパッドの右上
KEY_B2	キーパッドの中央
KEY_C1	キーパッドの左下
KEY_C3	キーパッドの右下
KEY_BTAB	バックタブキー
KEY_BEG	開始 (Beginning) キー
KEY_CANCEL	キャンセルキー
KEY_CLOSE	クローズキー
KEY_COMMAND	コマンド (cmd) キー
KEY_COPY	コピーキー
KEY_CREATE	作成キー
KEY_END	終了キー
KEY_EXIT	Exit キー
KEY_FIND	検索キー
KEY_HELP	ヘルプキー
KEY_MARK	マークキー
KEY_MESSAGE	メッセージキー
KEY_MOVE	移動キー
KEY_NEXT	次オブジェクトキー
KEY_OPEN	オープンキー
KEY_OPTIONS	オプションキー
KEY_PREVIOUS	前オブジェクトキー
KEY_REDO	再実行キー
KEY_REFERENCE	参照キー
KEY_REFRESH	リフレッシュキー

名前	キー名
KEY_REPLACE	置換キー
KEY_RESTART	再スタートキー
KEY_RESUME	再開キー
KEY_SAVE	セーブキー
KEY_SBEG	シフト付き開始キー
KEY_SCANCEL	シフト付きキャンセルキー
KEY_SCOMMAND	シフト付きコマンドキー
KEY_SCOPY	シフト付きコピーキー
KEY_SCREATE	シフト付き作成キー
KEY_SDC	シフト付き文字削除キー
KEY_SDL	シフト付き行削除キー
KEY_SELECT	選択キー
KEY_SEND	シフト付き送信キー
KEY_SEOL	シフト付き行クリアキー
KEY_SEXIT	シフト付き Exit キー
KEY_SFIND	シフト付き検索キー
KEY_SHELP	シフト付きヘルプキー
KEY_SHOME	シフト付き Home キー
KEY_SIC	シフト付き入力キー
KEY_SLEFT	シフト付き左向き矢印キー
KEY_SMESSAGE	シフト付きメッセージキー
KEY_SMOVE	シフト付き移動キー
KEY_SNEXT	シフト付き次オブジェクトキー
KEY_SOPTIONS	シフト付きオプションキー
KEY_SPREVIOUS	シフト付き前オブジェクトキー
KEY_SPRINT	シフト付き印刷キー
KEY_SREDO	シフト付き再実行キー
KEY_SREPLACE	シフト付き置換キー
KEY_SRIGHT	シフト付き右向き矢印キー

名前	キー名
KEY_SRSUME	シフト付き再開キー
KEY_SSAVE	シフト付きセーブキー
KEY_SSUSPEND	シフト付き中断キー
KEY_SUNDO	シフト付き取消キー
KEY_SUSPEND	中断キー
KEY_UNDO	取消キー

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目　curses(3CURSES), curs_inopts(3CURSES), curs_move(3CURSES), wrefresh(3CURSES), terminfo(4), attributes(5)

注意事項　ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

単一文字関数に対してエスケープ文字を使うことはできません。

getwch()、wgetwch()、mvgetwch()、mvwgetwch() のいずれかの関数を用いる際、nocbreak モードと echo モードを同時に使用することはできません。個々の文字が入力されたときの tty ドライバの状態によっては、プログラムは予測したものとは異なる結果を生成する場合があります。

getwch()、mvgetwch()、mvwgetwch() の各ルーチンはマクロにすることも可能です。

名前 curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の wchar_t 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^*X* 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inswch, inswch, winswch, mvinswch, mvwinswch – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字を挿入

形式 |
```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswch(chtype wch);

int winswch(WINDOW *win, chtype wch);

int mvinswch(int y, int x, chtype wch);

int mvwinswch(WINDOW *win, int y, int x, chtype wch);
```

機能説明 | これらのルーチンは、wchar_t 文字を含んでいる *wch* を、現カーソル位置の文字の直前に挿入します。これによりカーソルの右側にある文字はそれぞれ 1 文字分ずつ右に移動し、その結果、行の右端にある文字が失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用する ということを意味するものではありません。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

inswch()、mvinswch()、mvwinswch() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^X 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inwch, inwch, winwch, mvinwch, mvwinwch – curses ウィンドウから wchar_t 文字とその属性を得る

形式 |
```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

chtype inwch(void);

chtype winwch(WINDOW *win);

chtype mvinwch(int y, int x);

chtype mvwinwch(WINDOW *win, int y, int x);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置にある、chtype タイプの wchar_t 文字を返します。その位置に属性が設定されていれば、それらの属性値の論理和も返されます。<curses.h> 中に定義されている定数を & (論理積) 演算子とともに用いて、文字だけもしくは属性だけを抽出することも可能です。

属性 | winwch() が返す値と以下に示すビットマスクとの論理積をとることもできます。

A_WCHARTEXT	文字抽出用のビットマスク
A_WATTRIBUTES	属性抽出用のビットマスク

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

全ルーチンともマクロにすることが可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の `wchar_t` 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の `wchar_t` 文字を返します。

戻り値 上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性 次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 `curses`(3CURSES), `attributes`(5)

注意事項 ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`winnwstr()` 以外のルーチンはマクロにすることも可能です。

名前

curs_alecompat, movenextch, wmovenextch, moveprevch, wmoveprevch, adjcurspos, wadjcurspos – カーソルを文字単位に移動するために ALE curses ライブラリに追加された関数

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int movenextch(void);

int wmovenextch(WINDOW *win);

int moveprevch(void);

int wmoveprevch(WINDOW *win);

int adjcurspos(void);

int wadjcurspos(WINDOW *win);
```

機能説明

上記 movenextch() および wmovenextch() ルーチンは、カーソルを次の (右方の) 文字に移動します。次の文字が複数カラム文字であれば、新カーソル位置はその文字の最初の (左端の) カラムとなります。現カーソル位置が複数カラム文字の左端カラムであっても、新しい位置は次の文字上に移動します。なお現カーソル位置が複数カラム文字上の場合、単純なカーソル移動 (++x) を実行しても次の文字に移動するとは保証できません。新しいカーソル位置は getyx(3CURSES) を実行することにより得られます。

moveprevc() および wmoveprevch() ルーチンは、上記 movenextch() と wmovenextch() とは反対方向に、つまり 1 つ前の文字の左端カラムにカーソルを移動します。

adjcurspos() および wadjcurspos() ルーチンは、現在カーソルが置かれている複数カラム文字の左端のカラムへカーソルを移動します。現在の位置がすでに左端カラムである場合、および現在置かれている文字がシングルカラム文字の場合、これらのルーチンは意味を持たず、カーソル位置は変わりません。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getyx(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

movenextch()、moveprevch()、adjcurspos() の各ルーチンはマクロにすることも可能です。

名前 | curs_alecompat, movenextch, wmovenextch, moveprevch, wmoveprevch, adjcurspos, wadjcurspos – カーソルを文字単位に移動するために ALE curses ライブラリに追加された関数

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int movenextch(void);

int wmovenextch(WINDOW *win);

int moveprevch(void);

int wmoveprevch(WINDOW *win);

int adjcurspos(void);

int wadjcurspos(WINDOW *win);
```

機能説明 | 上記 movenextch() および wmovenextch() ルーチンは、カーソルを次の (右方の) 文字に移動します。次の文字が複数カラム文字であれば、新カーソル位置はその文字の最初の (左端の) カラムとなります。現カーソル位置が複数カラム文字の左端カラムであっても、新しい位置は次の文字上に移動します。なお現カーソル位置が複数カラム文字上の場合、単純なカーソル移動 (++x) を実行しても次の文字に移動するとは保証できません。新しいカーソル位置は getyx(3CURSES) を実行することにより得られます。

moveprevc() および wmoveprevch() ルーチンは、上記 movenextch() と wmovenextch() とは反対方向に、つまり 1 つ前の文字の左端カラムにカーソルを移動します。

adjcurspos() および wadjcurspos() ルーチンは、現在カーソルが置かれている複数カラム文字の左端のカラムへカーソルを移動します。現在の位置がすでに左端カラムである場合、および現在置かれている文字がシングルカラム文字の場合、これらのルーチンは意味を持たず、カーソル位置は変わりません。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), getyx(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

movenextch()、moveprevch()、adjcurspos() の各ルーチンはマクロにすることも可能です。

名前 | curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式 | **cc** [*flag*...] *file*... -lcurses [*library*...]

```
#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | 上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_addwch, addwch, waddwch, mvaddwch, mvwaddwch, echowchar, wechowchar – wchar_t 文字を属性とともに curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwch(chtype wch);

int waddwch(WINDOW *win, chtype wch);

int mvaddwch(int y, int x, chtype wch);

int mvwaddwch(WINDOW *win, int y, int x, chtype wch);

int echowchar(chtype wch);

int wechowchar(WINDOW *win, chtype wch);
```

機能説明

addwch()、waddwch()、mvaddwch()、mvwaddwch() の各ルーチンは、wchar_t を保ちながら *wch* 文字をウィンドウ中の現カーソル位置に書き出し、ウィンドウカーソルの位置を進めます。この機能は C 複数バイトライブラリ内にある putwchar(3C) の機能に似ています。右側のマージンでは自動的に復帰改行 (newline) が行われます。scrollok が有効であれば、スクロール領域が上に 1 行スクロールされます。

wch がタブ、復帰改行、バックスペースのいずれかの場合、それに従ってウィンドウ内でカーソルが移動します。なお復帰改行の場合には、カーソル移動に先だって clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wch* が他の制御文字の場合、^*X* 形式で描かれます。制御文字を追加した後で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

ビデオ属性は、論理和をパラメータ中にとることにより wchar_t 文字と組み合わせることができます。その結果、属性の設定も行われます。これが意味することは、inwch() と addwch() を使ってテキスト (属性も含む) をある場所から他の場所へコピーすることが可能であるということです。standout(3CURSES) の、定義済みビデオ属性定数の項を参照してください。

機能的に見ると、echowchar() ルーチンは addwch() と refresh(3CURSES) を連続して呼び出すことと同等で、wechowchar() ルーチンは waddwch() と wrefresh(3CURSES) を連続して呼び出すことと同等です。両ルーチンは 1 文字だけしか出力されないことが判っていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。

線グラフィック

addwch() ファミリのルーチンとともに以下の変数を使用して、線を描画する文字を画面に書き出すことができます。端末に変数が定義されているとき、A_ALTCHARSET ビットがオンになります (curs_attr(3CURSES) の項を参照)。変数が定義されていなければ、以下に述べるデフォルト文字が変数に与えられます。なお各変数の名前は VT100 命名規約に準拠しています。

名前	デフォルト	グリフ記述
ACS_ULCORNER	+	左上角
ACS_LLCORNER	+	左下角
ACS_URCORNER	+	右上角
ACS_LRCORNER	+	右下角
ACS_RTEE	+	右端
ACS_LTEE	+	左端
ACS_BTEE	+	下端
ACS_TTEE	+	上端
ACS_HLINE	–	横線
ACS_VLINE	|	縦線
ACS_PLUS	+	正符号
ACS_S1	-	スキャンライン 1
ACS_S9	_	スキャンライン 9
ACS_DIAMOND	+	ダイヤモンド
ACS_CKBOARD	:	市松模様 (点描)
ACS_DEGREE	'	度数記号
ACS_PLMINUS	#	プラス／マイナス
ACS_BULLET	o	丸
ACS_LARROW	<	左向きの矢印
ACS_RARROW	>	右向きの矢印
ACS_DARROW	v	下向きの矢印
ACS_UARROW	^	上向きの矢印
ACS_BOARD	#	正方形のボード
ACS_LANTERN	#	ランタン記号
ACS_BLOCK	#	正方形のブロック

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 putwchar(3C), clrtoeol(3CURSES), curses(3CURSES), curs_attr(3CURSES), curs_inwch(3CURSES), curs_outopts(3CURSES), refresh(3CURSES), standout(3CURSES), winwch(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

addwch()、mvaddwch()、mvwaddwch()、echowchar() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明

上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

`waddwchnstr()` 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、`chtype` 中でカラー属性を扱うことはできません。

名前

curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明

上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwchnstr() 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式 | **cc** [*flag*...] *file*... -lcurses [*library*...]

```
#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | 上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_getwch, getwch, wgetwch, mvgetwch, mvwgetwch, ungetwch – curses 端末キーボードから wchar_t 文字を得る (または戻す)

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwch(void);

int wgetwch(WINDOW *win);

int mvgetwch(int y, int x);

int mvwgetwch(WINDOW *win, int y, int x);

int ungetwch(int wch);
```

機能説明

getwch()、wgetwch()、mvgetwch()、mvwgetwch() の各ルーチンは、ウィンドウに対応づけられた端末から EUC 文字 1 文字を読み込み、それを wchar_t 文字に転送し、wchar_t 文字を返します。遅延無しモードでは、入力待ちのものがなければ ERR 値が返されます。遅延モードでは、システムからテキストが渡されるまでプログラムは待ち状態に置かれます。待つタイミングは cbreak の設定により異なり、1 つの文字の後 (cbreak モード) または最初の改行の後 (nocbreak モード) となります。また半遅延モードでは、何らかの文字が入力されるまであるいは指定されたタイムアウト値に達するまで待ちます。noecho が設定されていなければ、読み込まれた文字は指定されたウィンドウ中にエコーされます。

ウィンドウがパッドでなく、最後に wrefresh(3CURSES) が呼び出されてからウィンドウの移動または変更が行われていれば、次の文字を読み込む前に wrefresh が呼び出されます。

keypad が TRUE の場合、ファンクションキーが押されると実際の文字の代わりにファンクションキーのトークンが返されます。使用可能なファンクションキーは KEY_ で始まる名前で、0401 で始まる整数とともに <curses.h> 中に定義されています。ファンクションキーの先頭文字でありうる文字 (たとえばエスケープ文字) を受け取ると、curses(3CURSES) はタイマをセットします。指定された時間内にそのシーケンスの残りの文字が到着しなければ、受け取った 1 文字だけを渡します。時間内に残りの文字を受け取れば、ファンクションキーの値を返します。このため、エスケープ文字を押したのにそれがプログラムに渡されるまでに時間がかかった、 というケースを多くの端末が経験することになります。

ungetwch() ルーチンは、次に wgetwch() を呼び出したときに返される入力キュー中に *wch* 文字を戻します。

ファンクションキー

keypad が有効であれば、以下の表にリストされているファンクションキー (いずれも <curses.h> 中に定義) が getwch() により返されます。なお端末によっては、これらすべてのファンクションキーがサポートされるとは限りません。サポートされないケースは、そのキーが押されたときにその端末転送するコードが一意ではない場合、およびそのキーの定義が terminfo(4) データベース中に存在しない場合です。

名前	キー名
KEY_BREAK	Break キー
KEY_DOWN	4 つの矢印キー ...
KEY_UP	
KEY_LEFT	
KEY_RIGHT	
KEY_HOME	Home キー (左上向き矢印)
KEY_BACKSPACE	バックスペース
KEY_F0	ファンクションキー (64 個分の空白が予約されている)
KEY_F(*n*)	For $0 \leq n \leq 63$
KEY_DL	行削除
KEY_IL	行挿入
KEY_DC	文字削除
KEY_IC	文字挿入、または挿入モードに入る
KEY_EIC	挿入モードを抜ける
KEY_CLEAR	画面クリア
KEY_EOS	画面の終わりまでクリア
KEY_EOL	行の終わりまでクリア
KEY_SF	上方へ 1 行スクロール
KEY_SR	下方へ 1 行スクロール (リバース)
KEY_NPAGE	次ページ
KEY_PPAGE	前ページ
KEY_STAB	タブを設定
KEY_CTAB	タブをクリア
KEY_CATAB	全タブをクリア
KEY_ENTER	入力または送信
KEY_SRESET	ソフト (部分的) リセット
KEY_RESET	リセットまたはハードリセット
KEY_PRINT	印刷またはコピー

名前	キー名
KEY_LL	下部のホーム位置 (左下) へ。 キーパッドの割り当ては次のとおり。 A1 up A3 left B2 right C1 down C3
KEY_A1	キーパッドの左上
KEY_A3	キーパッドの右上
KEY_B2	キーパッドの中央
KEY_C1	キーパッドの左下
KEY_C3	キーパッドの右下
KEY_BTAB	バックタブキー
KEY_BEG	開始 (Beginning) キー
KEY_CANCEL	キャンセルキー
KEY_CLOSE	クローズキー
KEY_COMMAND	コマンド (cmd) キー
KEY_COPY	コピーキー
KEY_CREATE	作成キー
KEY_END	終了キー
KEY_EXIT	Exit キー
KEY_FIND	検索キー
KEY_HELP	ヘルプキー
KEY_MARK	マークキー
KEY_MESSAGE	メッセージキー
KEY_MOVE	移動キー
KEY_NEXT	次オブジェクトキー
KEY_OPEN	オープンキー
KEY_OPTIONS	オプションキー
KEY_PREVIOUS	前オブジェクトキー
KEY_REDO	再実行キー
KEY_REFERENCE	参照キー
KEY_REFRESH	リフレッシュキー

名前	キー名
KEY_REPLACE	置換キー
KEY_RESTART	再スタートキー
KEY_RESUME	再開キー
KEY_SAVE	セーブキー
KEY_SBEG	シフト付き開始キー
KEY_SCANCEL	シフト付きキャンセルキー
KEY_SCOMMAND	シフト付きコマンドキー
KEY_SCOPY	シフト付きコピーキー
KEY_SCREATE	シフト付き作成キー
KEY_SDC	シフト付き文字削除キー
KEY_SDL	シフト付き行削除キー
KEY_SELECT	選択キー
KEY_SEND	シフト付き送信キー
KEY_SEOL	シフト付き行クリアキー
KEY_SEXIT	シフト付き Exit キー
KEY_SFIND	シフト付き検索キー
KEY_SHELP	シフト付きヘルプキー
KEY_SHOME	シフト付き Home キー
KEY_SIC	シフト付き入力キー
KEY_SLEFT	シフト付き左向き矢印キー
KEY_SMESSAGE	シフト付きメッセージキー
KEY_SMOVE	シフト付き移動キー
KEY_SNEXT	シフト付き次オブジェクトキー
KEY_SOPTIONS	シフト付きオプションキー
KEY_SPREVIOUS	シフト付き前オブジェクトキー
KEY_SPRINT	シフト付き印刷キー
KEY_SREDO	シフト付き再実行キー
KEY_SREPLACE	シフト付き置換キー
KEY_SRIGHT	シフト付き右向き矢印キー

名前	キー名
KEY_SRSUME	シフト付き再開キー
KEY_SSAVE	シフト付きセーブキー
KEY_SSUSPEND	シフト付き中断キー
KEY_SUNDO	シフト付き取消キー
KEY_SUSPEND	中断キー
KEY_UNDO	取消キー

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目　curses(3CURSES), curs_inopts(3CURSES), curs_move(3CURSES), wrefresh(3CURSES), terminfo(4), attributes(5)

注意事項　ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

単一文字関数に対してエスケープ文字を使うことはできません。

getwch()、wgetwch()、mvgetwch()、mvwgetwch() のいずれかの関数を用いる際、nocbreak モードと echo モードを同時に使用することはできません。個々の文字が入力されたときの tty ドライバの状態によっては、プログラムは予測したものとは異なる結果を生成する場合があります。

getwch()、mvgetwch()、mvwgetwch() の各ルーチンはマクロにすることも可能です。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の wchar_t 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式 |

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^*X* 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inswch, inswch, winswch, mvinswch, mvwinswch – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswch(chtype wch);

int winswch(WINDOW *win, chtype wch);

int mvinswch(int y, int x, chtype wch);

int mvwinswch(WINDOW *win, int y, int x, chtype wch);
```

機能説明 | これらのルーチンは、wchar_t 文字を含んでいる *wch* を、現カーソル位置の文字の直前に挿入します。これによりカーソルの右側にある文字はそれぞれ 1 文字分ずつ右に移動し、その結果、行の右端にある文字が失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用する ということを意味するものではありません。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

inswch()、mvinswch()、mvwinswch() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^X 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_inwch, inwch, winwch, mvinwch, mvwinwch – curses ウィンドウから wchar_t 文字とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

chtype inwch(void);

chtype winwch(WINDOW *win);

chtype mvinwch(int y, int x);

chtype mvwinwch(WINDOW *win, int y, int x);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置にある、chtype タイプの wchar_t 文字を返します。その位置に属性が設定されていれば、それらの属性値の論理和も返されます。<curses.h> 中に定義されている定数を & (論理積) 演算子とともに用いて、文字だけもしくは属性だけを抽出することも可能です。

属性

winwch() が返す値と以下に示すビットマスクとの論理積をとることもできます。

A_WCHARTEXT	文字抽出用のビットマスク
A_WATTRIBUTES	属性抽出用のビットマスク

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

全ルーチンともマクロにすることが可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式 |

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式 |

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の `wchar_t` 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の `wchar_t` 文字を返します。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性 | 次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | `curses`(3CURSES), `attributes`(5)

注意事項 | ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`winnwstr()` 以外のルーチンはマクロにすることも可能です。

名前 | curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式 | **cc** [*flag*...] *file*... -lcurses [*library*...]

```
#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | 上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_addwch, addwch, waddwch, mvaddwch, mvwaddwch, echowchar, wechowchar – wchar_t 文字を属性とともに curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwch(chtype wch);

int waddwch(WINDOW *win, chtype wch);

int mvaddwch(int y, int x, chtype wch);

int mvwaddwch(WINDOW *win, int y, int x, chtype wch);

int echowchar(chtype wch);

int wechowchar(WINDOW *win, chtype wch);
```

機能説明

addwch()、waddwch()、mvaddwch()、mvwaddwch() の各ルーチンは、wchar_t を保ちながら *wch* 文字をウィンドウ中の現カーソル位置に書き出し、ウィンドウカーソルの位置を進めます。この機能は C 複数バイトライブラリ内にある putwchar(3C) の機能に似ています。右側のマージンでは自動的に復帰改行 (newline) が行われます。scrollok が有効であれば、スクロール領域が上に 1 行スクロールされます。

wch がタブ、復帰改行、バックスペースのいずれかの場合、それに従ってウィンドウ内でカーソルが移動します。なお復帰改行の場合には、カーソル移動に先だって clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wch* が他の制御文字の場合、^*X* 形式で描かれます。制御文字を追加した後で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

ビデオ属性は、論理和をパラメータ中にとることにより wchar_t 文字と組み合わせることができます。その結果、属性の設定も行われます。これが意味することは、inwch() と addwch() を使ってテキスト (属性も含む) をある場所から他の場所へコピーすることが可能であるということです。standout(3CURSES) の、定義済みビデオ属性定数の項を参照してください。

機能的に見ると、echowchar() ルーチンは addwch() と refresh(3CURSES) を連続して呼び出すことと同等で、wechowchar() ルーチンは waddwch() と wrefresh(3CURSES) を連続して呼び出すことと同等です。両ルーチンは 1 文字だけしか出力されないことが判っていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。

線グラフィック

addwch() ファミリのルーチンとともに以下の変数を使用して、線を描画する文字を画面に書き出すことができます。端末に変数が定義されているとき、A_ALTCHARSET ビットがオンになります (curs_attr(3CURSES) の項を参照)。変数が定義されていなければ、以下に述べるデフォルト文字が変数に与えられます。なお各変数の名前は VT100 命名規約に準拠しています。

名前	デフォルト	グリフ記述
ACS_ULCORNER	+	左上角
ACS_LLCORNER	+	左下角
ACS_URCORNER	+	右上角
ACS_LRCORNER	+	右下角
ACS_RTEE	+	右端
ACS_LTEE	+	左端
ACS_BTEE	+	下端
ACS_TTEE	+	上端
ACS_HLINE	–	横線
ACS_VLINE	|	縦線
ACS_PLUS	+	正符号
ACS_S1	-	スキャンライン 1
ACS_S9	_	スキャンライン 9
ACS_DIAMOND	+	ダイヤモンド
ACS_CKBOARD	:	市松模様 (点描)
ACS_DEGREE	'	度数記号
ACS_PLMINUS	#	プラス／マイナス
ACS_BULLET	o	丸
ACS_LARROW	<	左向きの矢印
ACS_RARROW	>	右向きの矢印
ACS_DARROW	v	下向きの矢印
ACS_UARROW	^	上向きの矢印
ACS_BOARD	#	正方形のボード
ACS_LANTERN	#	ランタン記号
ACS_BLOCK	#	正方形のブロック

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 putwchar(3C), clrtoeol(3CURSES), curses(3CURSES), curs_attr(3CURSES), curs_inwch(3CURSES), curs_outopts(3CURSES), refresh(3CURSES), standout(3CURSES), winwch(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

addwch()、mvaddwch()、mvwaddwch()、echowchar() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明

上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

`waddwchnstr()` 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、`chtype` 中でカラー属性を扱うことはできません。

名前

curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明

上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwchnstr() 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式 | **cc** [*flag*...] *file*... -lcurses [*library*...]

```
#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | 上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_getwch, getwch, wgetwch, mvgetwch, mvwgetwch, ungetwch – curses 端末キーボードから wchar_t 文字を得る (または戻す)

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwch(void);

int wgetwch(WINDOW *win);

int mvgetwch(int y, int x);

int mvwgetwch(WINDOW *win, int y, int x);

int ungetwch(int wch);
```

機能説明

getwch()、wgetwch()、mvgetwch()、mvwgetwch() の各ルーチンは、ウィンドウに対応づけられた端末から EUC 文字 1 文字を読み込み、それを wchar_t 文字に転送し、wchar_t 文字を返します。遅延無しモードでは、入力待ちのものがなければ ERR 値が返されます。遅延モードでは、システムからテキストが渡されるまでプログラムは待ち状態に置かれます。待つタイミングは cbreak の設定により異なり、1 つの文字の後 (cbreak モード) または最初の改行の後 (nocbreak モード) となります。また半遅延モードでは、何らかの文字が入力されるまであるいは指定されたタイムアウト値に達するまで待ちます。noecho が設定されていなければ、読み込まれた文字は指定されたウィンドウ中にエコーされます。

ウィンドウがパッドでなく、最後に wrefresh(3CURSES) が呼び出されてからウィンドウの移動または変更が行われていれば、次の文字を読み込む前に wrefresh が呼び出されます。

keypad が TRUE の場合、ファンクションキーが押されると実際の文字の代わりにファンクションキーのトークンが返されます。使用可能なファンクションキーは KEY_ で始まる名前で、0401 で始まる整数とともに <curses.h> 中に定義されています。ファンクションキーの先頭文字でありうる文字 (たとえばエスケープ文字) を受け取ると、curses(3CURSES) はタイマをセットします。指定された時間内にそのシーケンスの残りの文字が到着しなければ、受け取った 1 文字だけを渡します。時間内に残りの文字を受け取れば、ファンクションキーの値を返します。このため、エスケープ文字を押したのにそれがプログラムに渡されるまでに時間がかかった、 というケースを多くの端末が経験することになります。

ungetwch() ルーチンは、次に wgetwch() を呼び出したときに返される入力キュー中に *wch* 文字を戻します。

ファンクションキー

keypad が有効であれば、以下の表にリストされているファンクションキー (いずれも <curses.h> 中に定義) が getwch() により返されます。なお端末によっては、これらすべてのファンクションキーがサポートされるとは限りません。サポートされないケースは、そのキーが押されたときにその端末転送するコードが一意ではない場合、およびそのキーの定義が terminfo(4) データベース中に存在しない場合です。

名前	キー名
KEY_BREAK	Break キー
KEY_DOWN	4 つの矢印キー ...
KEY_UP	
KEY_LEFT	
KEY_RIGHT	
KEY_HOME	Home キー (左上向き矢印)
KEY_BACKSPACE	バックスペース
KEY_F0	ファンクションキー (64 個分の空白が予約されている)
KEY_F(*n*)	For $0 \leq n \leq 63$
KEY_DL	行削除
KEY_IL	行挿入
KEY_DC	文字削除
KEY_IC	文字挿入、または挿入モードに入る
KEY_EIC	挿入モードを抜ける
KEY_CLEAR	画面クリア
KEY_EOS	画面の終わりまでクリア
KEY_EOL	行の終わりまでクリア
KEY_SF	上方へ 1 行スクロール
KEY_SR	下方へ 1 行スクロール (リバース)
KEY_NPAGE	次ページ
KEY_PPAGE	前ページ
KEY_STAB	タブを設定
KEY_CTAB	タブをクリア
KEY_CATAB	全タブをクリア
KEY_ENTER	入力または送信
KEY_SRESET	ソフト (部分的) リセット
KEY_RESET	リセットまたはハードリセット
KEY_PRINT	印刷またはコピー

名前	キー名
KEY_LL	下部のホーム位置 (左下) へ。 キーパッドの割り当ては次のとおり。 A1 up A3 left B2 right C1 down C3
KEY_A1	キーパッドの左上
KEY_A3	キーパッドの右上
KEY_B2	キーパッドの中央
KEY_C1	キーパッドの左下
KEY_C3	キーパッドの右下
KEY_BTAB	バックタブキー
KEY_BEG	開始 (Beginning) キー
KEY_CANCEL	キャンセルキー
KEY_CLOSE	クローズキー
KEY_COMMAND	コマンド (cmd) キー
KEY_COPY	コピーキー
KEY_CREATE	作成キー
KEY_END	終了キー
KEY_EXIT	Exit キー
KEY_FIND	検索キー
KEY_HELP	ヘルプキー
KEY_MARK	マークキー
KEY_MESSAGE	メッセージキー
KEY_MOVE	移動キー
KEY_NEXT	次オブジェクトキー
KEY_OPEN	オープンキー
KEY_OPTIONS	オプションキー
KEY_PREVIOUS	前オブジェクトキー
KEY_REDO	再実行キー
KEY_REFERENCE	参照キー
KEY_REFRESH	リフレッシュキー

名前	キー名
KEY_REPLACE	置換キー
KEY_RESTART	再スタートキー
KEY_RESUME	再開キー
KEY_SAVE	セーブキー
KEY_SBEG	シフト付き開始キー
KEY_SCANCEL	シフト付きキャンセルキー
KEY_SCOMMAND	シフト付きコマンドキー
KEY_SCOPY	シフト付きコピーキー
KEY_SCREATE	シフト付き作成キー
KEY_SDC	シフト付き文字削除キー
KEY_SDL	シフト付き行削除キー
KEY_SELECT	選択キー
KEY_SEND	シフト付き送信キー
KEY_SEOL	シフト付き行クリアキー
KEY_SEXIT	シフト付き Exit キー
KEY_SFIND	シフト付き検索キー
KEY_SHELP	シフト付きヘルプキー
KEY_SHOME	シフト付き Home キー
KEY_SIC	シフト付き入力キー
KEY_SLEFT	シフト付き左向き矢印キー
KEY_SMESSAGE	シフト付きメッセージキー
KEY_SMOVE	シフト付き移動キー
KEY_SNEXT	シフト付き次オブジェクトキー
KEY_SOPTIONS	シフト付きオプションキー
KEY_SPREVIOUS	シフト付き前オブジェクトキー
KEY_SPRINT	シフト付き印刷キー
KEY_SREDO	シフト付き再実行キー
KEY_SREPLACE	シフト付き置換キー
KEY_SRIGHT	シフト付き右向き矢印キー

名前	キー名
KEY_SRSUME	シフト付き再開キー
KEY_SSAVE	シフト付きセーブキー
KEY_SSUSPEND	シフト付き中断キー
KEY_SUNDO	シフト付き取消キー
KEY_SUSPEND	中断キー
KEY_UNDO	取消キー

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目　curses(3CURSES), curs_inopts(3CURSES), curs_move(3CURSES), wrefresh(3CURSES), terminfo(4), attributes(5)

注意事項　ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

単一文字関数に対してエスケープ文字を使うことはできません。

getwch()、wgetwch()、mvgetwch()、mvwgetwch() のいずれかの関数を用いる際、nocbreak モードと echo モードを同時に使用することはできません。個々の文字が入力されたときの tty ドライバの状態によっては、プログラムは予測したものとは異なる結果を生成する場合があります。

getwch()、mvgetwch()、mvwgetwch() の各ルーチンはマクロにすることも可能です。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の `wchar_t` 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の `wchar_t` 文字を返します。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性

次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

`curses`(3CURSES), `attributes`(5)

注意事項

ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`winnwstr()` 以外のルーチンはマクロにすることも可能です。

名前 | curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式 |

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^*X* 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inswch, inswch, winswch, mvinswch, mvwinswch – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswch(chtype wch);

int winswch(WINDOW *win, chtype wch);

int mvinswch(int y, int x, chtype wch);

int mvwinswch(WINDOW *win, int y, int x, chtype wch);
```

機能説明 | これらのルーチンは、wchar_t 文字を含んでいる *wch* を、現カーソル位置の文字の直前に挿入します。これによりカーソルの右側にある文字はそれぞれ 1 文字分ずつ右に移動し、その結果、行の右端にある文字が失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用する ということを意味するものではありません。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

inswch()、mvinswch()、mvwinswch() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^X 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inwch, inwch, winwch, mvinwch, mvwinwch – curses ウィンドウから wchar_t 文字とその属性を得る

形式 |
```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

chtype inwch(void);

chtype winwch(WINDOW *win);

chtype mvinwch(int y, int x);

chtype mvwinwch(WINDOW *win, int y, int x);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置にある、chtype タイプの wchar_t 文字を返します。その位置に属性が設定されていれば、それらの属性値の論理和も返されます。<curses.h> 中に定義されている定数を & (論理積) 演算子とともに用いて、文字だけもしくは属性だけを抽出することも可能です。

属性 | winwch() が返す値と以下に示すビットマスクとの論理積をとることもできます。

A_WCHARTEXT	文字抽出用のビットマスク
A_WATTRIBUTES	属性抽出用のビットマスク

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

全ルーチンともマクロにすることが可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の `wchar_t` 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の `wchar_t` 文字を返します。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性 | 次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | `curses`(3CURSES), `attributes`(5)

注意事項 | ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`winnwstr()` 以外のルーチンはマクロにすることも可能です。

名前

curs_pad, newpad, subpad, prefresh, pnoutrefresh, pechochar, pechowchar – curses パッドの作成と表示

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);

WINDOW *subpad(WINDOW *orig, int nlines, int ncols, int begin_y, int
     begin_x);

int prefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow, int
     smincol, int smaxrow, int smaxcol);

int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow,
     int smincol, int smaxrow, int smaxcol);

int pechochar(WINDOW *pad, chtype ch);

int pechowchar(WINDOW *pad, chtype wch);
```

機能説明

newpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるパッドデータ構造体を作成し、その構造体へのポインタを返します。パッドはウィンドウに似ていますが、ウィンドウとは異なり、画面のサイズにより制限されたり 画面上の特別な部分に対応したりしているとは限りません。パッドは大きなウィンドウが必要なときに使うことができ、同時にウィンドウの一部だけが画面上に表示されます。パッドに対しては (入力のスクロールやエコーなどからの) 自動 リフレッシュ処理は行われません。引数に *pad* を指定して wrefresh(3CURSES) を呼び出すことはできませんので、リフレッシュを行いたければ prefresh() または pnoutrefresh() ルーチンを呼び出してください。これらのルーチンに対しては、表示対象とするパッドの部分および 表示に用いる画面の位置を指定するパラメータを与える必要があります。

subpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるサブウィンドウをパッド内に作成し、そのサブウィンドウへのポインタを返します。 画面上の座標値を用いる subwin(3CURSES) とは異なり、ウィンドウはパッド上の (*begin_x, begin_y*) に作成されます。また新ウィンドウは元のウィンドウ *orig* 内に置かれるので、一方のウィンドウで行った修正は他方のウィンドウにも適用されます。本ルーチン使用時には、prefresh() を呼び出す前に *orig* 上で touchwin(3CURSES) または touchline(3CURSES) を呼び出す必要が生じる場合がよくあります。

prefresh() と pnoutrefresh() の両ルーチンは、wrefresh(3CURSES) と wnoutrefresh(3CURSES) の両ルーチンと同等機能ですが、ウィンドウではなくパッドを処理対象とする点が異なります。これらを呼び出す際、パッドおよび画面のどの部分を対象とするかを指定するパラメータが必要となります。*pminrow* と *pmincol* は、パッド中に表示する四角形の左上角の位置を指定します。*sminrow*、*smincol*、*smaxrow*、*smaxcol* の各引数は、画面に表示する四角形の各辺の位置を指定します。四角形の大きさは常に同じなので、パッド中の四角形右下角の位置は画面座標値から自動的に計算されます。この 2 つの四角形は、ともにそれぞれの構造体の中に完全に納まっていなければなりません。*pminrow*、*pmincol*、*sminrow*、*smincol* の各引数が負の値のときはゼロと見なされます。

pechochar() ルーチンは、機能的には addch(3CURSES) と refresh(3CURSES)、waddch(3CURSES) と wrefresh(3CURSES)、または waddch(3CURSES) と prefresh() を、それぞれ連続して呼び出すことと同等です。本ルーチンは 1 文字だけしか出力されないことがわかっていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較して パフォーマンスの大幅な向上が期待できます。pechochar() の場合、画面上のパッドの最新の位置が prefresh() への引数として再利用されます。

戻り値

整数値を返すルーチンは、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

ポインタを返すルーチンは、エラーが発生すれば NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

addch(3CURSES), curses(3CURSES), refresh(3CURSES), subwin(3CURSES), touchline(3CURSES), touchwin(3CURSES), waddch(3CURSES), wnoutrefresh(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項

ヘッダー <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーを含みます。

pechochar() はマクロにすることも可能です。

名前

curs_pad, newpad, subpad, prefresh, pnoutrefresh, pechochar, pechowchar – curses パッドの作成と表示

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);

WINDOW *subpad(WINDOW *orig, int nlines, int ncols, int begin_y, int
     begin_x);

int prefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow, int
     smincol, int smaxrow, int smaxcol);

int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow,
     int smincol, int smaxrow, int smaxcol);

int pechochar(WINDOW *pad, chtype ch);

int pechowchar(WINDOW *pad, chtype wch);
```

機能説明

newpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるパッドデータ構造体を作成し、その構造体へのポインタを返します。パッドはウィンドウに似ていますが、ウィンドウとは異なり、画面のサイズにより制限されたり 画面上の特別な部分に対応したりしているとは限りません。パッドは大きなウィンドウが必要なときに使うことができ、同時にウィンドウの一部だけが画面上に表示されます。パッドに対しては (入力のスクロールやエコーなどからの) 自動 リフレッシュ処理は行われません。引数に *pad* を指定して wrefresh(3CURSES) を呼び出すことはできませんので、リフレッシュを行いたければ prefresh() または pnoutrefresh() ルーチンを呼び出してください。これらのルーチンに対しては、表示対象とするパッドの部分および 表示に用いる画面の位置を指定するパラメータを与える必要があります。

subpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるサブウィンドウをパッド内に作成し、そのサブウィンドウへのポインタを返します。 画面上の座標値を用いる subwin(3CURSES) とは異なり、ウィンドウはパッド上の (*begin_x, begin_y*) に作成されます。また新ウィンドウは元のウィンドウ *orig* 内に置かれるので、一方のウィンドウで行った修正は他方のウィンドウにも適用されます。本ルーチン使用時には、prefresh() を呼び出す前に *orig* 上で touchwin(3CURSES) または touchline(3CURSES) を呼び出す必要が生じる場合がよくあります。

prefresh() と pnoutrefresh() の両ルーチンは、wrefresh(3CURSES) と wnoutrefresh(3CURSES) の両ルーチンと同等機能ですが、ウィンドウではなくパッドを処理対象とする点が異なります。これらを呼び出す際、パッドおよび画面のどの部分を対象とするかを指定するパラメータが必要となります。*pminrow* と *pmincol* は、パッド中に表示する四角形の左上角の位置を指定します。*sminrow*、*smincol*、*smaxrow*、*smaxcol* の各引数は、画面に表示する四角形の各辺の位置を指定します。四角形の大きさは常に同じなので、パッド中の四角形右下角の位置は画面座標値から自動的に計算されます。この 2 つの四角形は、ともにそれぞれの構造体の中に完全に納まっていなければなりません。*pminrow*、*pmincol*、*sminrow*、*smincol* の各引数が負の値のときはゼロと見なされます。

pechochar() ルーチンは、機能的には addch(3CURSES) と refresh(3CURSES)、waddch(3CURSES) と wrefresh(3CURSES)、または waddch(3CURSES) と prefresh() を、それぞれ連続して呼び出すことと同等です。本ルーチンは 1 文字だけしか出力されないことがわかっていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較して パフォーマンスの大幅な向上が期待できます。pechochar() の場合、画面上のパッドの最新の位置が prefresh() への引数として再利用されます。

戻り値

整数値を返すルーチンは、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

ポインタを返すルーチンは、エラーが発生すれば NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

addch(3CURSES), curses(3CURSES), refresh(3CURSES), subwin(3CURSES), touchline(3CURSES), touchwin(3CURSES), waddch(3CURSES), wnoutrefresh(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項

ヘッダー <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーを含みます。

pechochar() はマクロにすることも可能です。

名前

curs_pad, newpad, subpad, prefresh, pnoutrefresh, pechochar, pechowchar – curses パッドの作成と表示

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);

WINDOW *subpad(WINDOW *orig, int nlines, int ncols, int begin_y, int
     begin_x);

int prefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow, int
     smincol, int smaxrow, int smaxcol);

int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow,
     int smincol, int smaxrow, int smaxcol);

int pechochar(WINDOW *pad, chtype ch);

int pechowchar(WINDOW *pad, chtype wch);
```

機能説明

newpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるパッドデータ構造体を作成し、その構造体へのポインタを返します。パッドはウィンドウに似ていますが、ウィンドウとは異なり、画面のサイズにより制限されたり 画面上の特別な部分に対応したりしているとは限りません。パッドは大きなウィンドウが必要なときに使うことができ、同時にウィンドウの一部だけが画面上に表示されます。パッドに対しては (入力のスクロールやエコーなどからの) 自動 リフレッシュ処理は行われません。引数に *pad* を指定して wrefresh(3CURSES) を呼び出すことはできませんので、リフレッシュを行いたければ prefresh() または pnoutrefresh() ルーチンを呼び出してください。これらのルーチンに対しては、表示対象とするパッドの部分および 表示に用いる画面の位置を指定するパラメータを与える必要があります。

subpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるサブウィンドウをパッド内に作成し、そのサブウィンドウへのポインタを返します。 画面上の座標値を用いる subwin(3CURSES) とは異なり、ウィンドウはパッド上の (*begin_x*, *begin_y*) に作成されます。また新ウィンドウは元のウィンドウ *orig* 内に置かれるので、一方のウィンドウで行った修正は他方のウィンドウにも適用されます。本ルーチン使用時には、prefresh() を呼び出す前に *orig* 上で touchwin(3CURSES) または touchline(3CURSES) を呼び出す必要が生じる場合がよくあります。

prefresh() と pnoutrefresh() の両ルーチンは、wrefresh(3CURSES) と wnoutrefresh(3CURSES) の両ルーチンと同等機能ですが、ウィンドウではなくパッドを処理対象とする点が異なります。これらを呼び出す際、パッドおよび画面のどの部分を対象とするかを指定するパラメータが必要となります。*pminrow* と *pmincol* は、パッド中に表示する四角形の左上角の位置を指定します。*sminrow*、*smincol*、*smaxrow*、*smaxcol* の各引数は、画面に表示する四角形の各辺の位置を指定します。四角形の大きさは常に同じなので、パッド中の四角形右下角の位置は画面座標値から自動的に計算されます。この 2 つの四角形は、ともにそれぞれの構造体の中に完全に納まっていなければなりません。*pminrow*、*pmincol*、*sminrow*、*smincol* の各引数が負の値のときはゼロと見なされます。

pechochar() ルーチンは、機能的には addch(3CURSES) と refresh(3CURSES)、waddch(3CURSES) と wrefresh(3CURSES)、または waddch(3CURSES) と prefresh() を、それぞれ連続して呼び出すことと同等です。本ルーチンは 1 文字だけしか出力されないことがわかっていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。pechochar() の場合、画面上のパッドの最新の位置が prefresh() への引数として再利用されます。

戻り値

整数値を返すルーチンは、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

ポインタを返すルーチンは、エラーが発生すれば NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

addch(3CURSES), curses(3CURSES), refresh(3CURSES), subwin(3CURSES), touchline(3CURSES), touchwin(3CURSES), waddch(3CURSES), wnoutrefresh(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項

ヘッダー <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーを含みます。

pechochar() はマクロにすることも可能です。

名前

curs_pad, newpad, subpad, prefresh, pnoutrefresh, pechochar, pechowchar – curses パッドの作成と表示

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);

WINDOW *subpad(WINDOW *orig, int nlines, int ncols, int begin_y, int
     begin_x);

int prefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow, int
     smincol, int smaxrow, int smaxcol);

int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow,
     int smincol, int smaxrow, int smaxcol);

int pechochar(WINDOW *pad, chtype ch);

int pechowchar(WINDOW *pad, chtype wch);
```

機能説明

newpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるパッドデータ構造体を作成し、その構造体へのポインタを返します。パッドはウィンドウに似ていますが、ウィンドウとは異なり、画面のサイズにより制限されたり 画面上の特別な部分に対応したりしているとは限りません。パッドは大きなウィンドウが必要なときに使うことができ、同時にウィンドウの一部だけが画面上に表示されます。パッドに対しては (入力のスクロールやエコーなどからの) 自動 リフレッシュ処理は行われません。引数に *pad* を指定して wrefresh(3CURSES) を呼び出すことはできませんので、リフレッシュを行いたければ prefresh() または pnoutrefresh() ルーチンを呼び出してください。これらのルーチンに対しては、表示対象とするパッドの部分および 表示に用いる画面の位置を指定するパラメータを与える必要があります。

subpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるサブウィンドウをパッド内に作成し、そのサブウィンドウへのポインタを返します。 画面上の座標値を用いる subwin(3CURSES) とは異なり、ウィンドウはパッド上の (*begin_x, begin_y*) に作成されます。また新ウィンドウは元のウィンドウ *orig* 内に置かれるので、一方のウィンドウで行った修正は他方のウィンドウにも適用されます。本ルーチン使用時には、prefresh() を呼び出す前に *orig* 上で touchwin(3CURSES) または touchline(3CURSES) を呼び出す必要が生じる場合がよくあります。

prefresh() と pnoutrefresh() の両ルーチンは、wrefresh(3CURSES) と wnoutrefresh(3CURSES) の両ルーチンと同等機能ですが、ウィンドウではなくパッドを処理対象とする点が異なります。これらを呼び出す際、パッドおよび画面のどの部分を対象とするかを指定するパラメータが必要となります。*pminrow* と *pmincol* は、パッド中に表示する四角形の左上角の位置を指定します。*sminrow*、*smincol*、*smaxrow*、*smaxcol* の各引数は、画面に表示する四角形の各辺の位置を指定します。四角形の大きさは常に同じなので、パッド中の四角形右下角の位置は画面座標値から自動的に計算されます。この 2 つの四角形は、ともにそれぞれの構造体の中に完全に納まっていなければなりません。*pminrow*、*pmincol*、*sminrow*、*smincol* の各引数が負の値のときはゼロと見なされます。

pechochar() ルーチンは、機能的には addch(3CURSES) と refresh(3CURSES)、waddch(3CURSES) と wrefresh(3CURSES)、または waddch(3CURSES) と prefresh() を、それぞれ連続して呼び出すことと同等です。本ルーチンは 1 文字だけしか出力されないことがわかっていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。pechochar() の場合、画面上のパッドの最新の位置が prefresh() への引数として再利用されます。

戻り値

整数値を返すルーチンは、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

ポインタを返すルーチンは、エラーが発生すれば NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

addch(3CURSES), curses(3CURSES), refresh(3CURSES), subwin(3CURSES), touchline(3CURSES), touchwin(3CURSES), waddch(3CURSES), wnoutrefresh(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項

ヘッダー <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーを含みます。

pechochar() はマクロにすることも可能です。

名前

curs_pad, newpad, subpad, prefresh, pnoutrefresh, pechochar, pechowchar – curses パッドの作成と表示

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);

WINDOW *subpad(WINDOW *orig, int nlines, int ncols, int begin_y, int
     begin_x);

int prefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow, int
     smincol, int smaxrow, int smaxcol);

int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow,
     int smincol, int smaxrow, int smaxcol);

int pechochar(WINDOW *pad, chtype ch);

int pechowchar(WINDOW *pad, chtype wch);
```

機能説明

newpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるパッドデータ構造体を作成し、その構造体へのポインタを返します。パッドはウィンドウに似ていますが、ウィンドウとは異なり、画面のサイズにより制限されたり 画面上の特別な部分に対応したりしているとは限りません。パッドは大きなウィンドウが必要なときに使うことができ、同時にウィンドウの一部だけが画面上に表示されます。パッドに対しては (入力のスクロールやエコーなどからの) 自動 リフレッシュ処理は行われません。引数に *pad* を指定して wrefresh(3CURSES) を呼び出すことはできませんので、リフレッシュを行いたければ prefresh() または pnoutrefresh() ルーチンを呼び出してください。これらのルーチンに対しては、表示対象とするパッドの部分および 表示に用いる画面の位置を指定するパラメータを与える必要があります。

subpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるサブウィンドウをパッド内に作成し、そのサブウィンドウへのポインタを返します。 画面上の座標値を用いる subwin(3CURSES) とは異なり、ウィンドウはパッド上の (*begin_x, begin_y*) に作成されます。また新ウィンドウは元のウィンドウ *orig* 内に置かれるので、一方のウィンドウで行った修正は他方のウィンドウにも適用されます。本ルーチン使用時には、prefresh() を呼び出す前に *orig* 上で touchwin(3CURSES) または touchline(3CURSES) を呼び出す必要が生じる場合がよくあります。

prefresh() と pnoutrefresh() の両ルーチンは、wrefresh(3CURSES) と wnoutrefresh(3CURSES) の両ルーチンと同等機能ですが、ウィンドウではなくパッドを処理対象とする点が異なります。これらを呼び出す際、パッドおよび画面のどの部分を対象とするかを指定するパラメータが必要となります。*pminrow* と *pmincol* は、パッド中に表示する四角形の左上角の位置を指定します。*sminrow*、*smincol*、*smaxrow*、*smaxcol* の各引数は、画面に表示する四角形の各辺の位置を指定します。四角形の大きさは常に同じなので、パッド中の四角形右下角の位置は画面座標値から自動的に計算されます。この 2 つの四角形は、ともにそれぞれの構造体の中に完全に納まっていなければなりません。*pminrow*、*pmincol*、*sminrow*、*smincol* の各引数が負の値のときはゼロと見なされます。

pechochar() ルーチンは、機能的には addch(3CURSES) と refresh(3CURSES)、waddch(3CURSES) と wrefresh(3CURSES)、または waddch(3CURSES) と prefresh() を、それぞれ連続して呼び出すことと同等です。本ルーチンは 1 文字だけしか出力されないことがわかっていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。pechochar() の場合、画面上のパッドの最新の位置が prefresh() への引数として再利用されます。

戻り値

整数値を返すルーチンは、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

ポインタを返すルーチンは、エラーが発生すれば NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

addch(3CURSES), curses(3CURSES), refresh(3CURSES), subwin(3CURSES), touchline(3CURSES), touchwin(3CURSES), waddch(3CURSES), wnoutrefresh(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項

ヘッダー <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーを含みます。

pechochar() はマクロにすることも可能です。

名前

curs_pad, newpad, subpad, prefresh, pnoutrefresh, pechochar, pechowchar – curses パッドの作成と表示

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

WINDOW *newpad(int nlines, int ncols);

WINDOW *subpad(WINDOW *orig, int nlines, int ncols, int begin_y, int
     begin_x);

int prefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow, int
     smincol, int smaxrow, int smaxcol);

int pnoutrefresh(WINDOW *pad, int pminrow, int pmincol, int sminrow,
     int smincol, int smaxrow, int smaxcol);

int pechochar(WINDOW *pad, chtype ch);

int pechowchar(WINDOW *pad, chtype wch);
```

機能説明

newpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるパッドデータ構造体を作成し、その構造体へのポインタを返します。パッドはウィンドウに似ていますが、ウィンドウとは異なり、画面のサイズにより制限されたり 画面上の特別な部分に対応したりしているとは限りません。パッドは大きなウィンドウが必要なときに使うことができ、同時にウィンドウの一部だけが画面上に表示されます。パッドに対しては (入力のスクロールやエコーなどからの) 自動 リフレッシュ処理は行われません。引数に *pad* を指定して wrefresh(3CURSES) を呼び出すことはできませんので、リフレッシュを行いたければ prefresh() または pnoutrefresh() ルーチンを呼び出してください。これらのルーチンに対しては、表示対象とするパッドの部分および 表示に用いる画面の位置を指定するパラメータを与える必要があります。

subpad() ルーチンは、*nlines* と *ncols* で指定された行数とカラム数からなるサブウィンドウをパッド内に作成し、そのサブウィンドウへのポインタを返します。 画面上の座標値を用いる subwin(3CURSES) とは異なり、ウィンドウはパッド上の (*begin_x, begin_y*) に作成されます。また新ウィンドウは元のウィンドウ *orig* 内に置かれるので、一方のウィンドウで行った修正は他方のウィンドウにも適用されます。本ルーチン使用時には、prefresh() を呼び出す前に *orig* 上で touchwin(3CURSES) または touchline(3CURSES) を呼び出す必要が生じる場合がよくあります。

prefresh() と pnoutrefresh() の両ルーチンは、wrefresh(3CURSES) と wnoutrefresh(3CURSES) の両ルーチンと同等機能ですが、ウィンドウではなくパッドを処理対象とする点が異なります。これらを呼び出す際、パッドおよび画面のどの部分を対象とするかを指定するパラメータが必要となります。*pminrow* と *pmincol* は、パッド中に表示する四角形の左上角の位置を指定します。*sminrow*、*smincol*、*smaxrow*、*smaxcol* の各引数は、画面に表示する四角形の各辺の位置を指定します。四角形の大きさは常に同じなので、パッド中の四角形右下角の位置は画面座標値から自動的に計算されます。この 2 つの四角形は、ともにそれぞれの構造体の中に完全に納まっていなければなりません。*pminrow*、*pmincol*、*sminrow*、*smincol* の各引数が負の値のときはゼロと見なされます。

pechochar() ルーチンは、機能的には addch(3CURSES) と refresh(3CURSES)、waddch(3CURSES) と wrefresh(3CURSES)、または waddch(3CURSES) と prefresh() を、それぞれ連続して呼び出すことと同等です。本ルーチンは 1 文字だけしか出力されないことがわかっていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較して パフォーマンスの大幅な向上が期待できます。pechochar() の場合、画面上のパッドの最新の位置が prefresh() への引数として再利用されます。

戻り値

整数値を返すルーチンは、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

ポインタを返すルーチンは、エラーが発生すれば NULL を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

addch(3CURSES), curses(3CURSES), refresh(3CURSES), subwin(3CURSES), touchline(3CURSES), touchwin(3CURSES), waddch(3CURSES), wnoutrefresh(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項

ヘッダー <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーを含みます。

pechochar() はマクロにすることも可能です。

名前

curs_getwch, getwch, wgetwch, mvgetwch, mvwgetwch, ungetwch – curses 端末キーボードから wchar_t 文字を得る (または戻す)

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwch(void);

int wgetwch(WINDOW *win);

int mvgetwch(int y, int x);

int mvwgetwch(WINDOW *win, int y, int x);

int ungetwch(int wch);
```

機能説明

getwch()、wgetwch()、mvgetwch()、mvwgetwch() の各ルーチンは、ウィンドウに対応づけられた端末から EUC 文字 1 文字を読み込み、それを wchar_t 文字に転送し、wchar_t 文字を返します。遅延無しモードでは、入力待ちのものがなければ ERR 値が返されます。遅延モードでは、システムからテキストが渡されるまでプログラムは待ち状態に置かれます。待つタイミングは cbreak の設定により異なり、1 つの文字の後 (cbreak モード) または最初の改行の後 (nocbreak モード) となります。また半遅延モードでは、何らかの文字が入力されるまであるいは指定されたタイムアウト値に達するまで待ちます。noecho が設定されていなければ、読み込まれた文字は指定されたウィンドウ中にエコーされます。

ウィンドウがパッドでなく、最後に wrefresh(3CURSES) が呼び出されてからウィンドウの移動または変更が行われていれば、次の文字を読み込む前に wrefresh が呼び出されます。

keypad が TRUE の場合、ファンクションキーが押されると実際の文字の代わりにファンクションキーのトークンが返されます。使用可能なファンクションキーは KEY_ で始まる名前で、0401 で始まる整数とともに <curses.h> 中に定義されています。ファンクションキーの先頭文字でありうる文字 (たとえばエスケープ文字) を受け取ると、curses(3CURSES) はタイマをセットします。指定された時間内にそのシーケンスの残りの文字が到着しなければ、受け取った 1 文字だけを渡します。時間内に残りの文字を受け取れば、ファンクションキーの値を返します。このため、エスケープ文字を押したのにそれがプログラムに渡されるまでに時間がかかった、 というケースを多くの端末が経験することになります。

ungetwch() ルーチンは、次に wgetwch() を呼び出したときに返される入力キュー中に *wch* 文字を戻します。

ファンクションキー

keypad が有効であれば、以下の表にリストされているファンクションキー (いずれも <curses.h> 中に定義) が getwch() により返されます。なお端末によっては、これらすべてのファンクションキーがサポートされるとは限りません。サポートされないケースは、そのキーが押されたときにその端末転送するコードが一意ではない場合、およびそのキーの定義が terminfo(4) データベース中に存在しない場合です。

名前	キー名
KEY_BREAK	Break キー
KEY_DOWN	4 つの矢印キー ...
KEY_UP	
KEY_LEFT	
KEY_RIGHT	
KEY_HOME	Home キー (左上向き矢印)
KEY_BACKSPACE	バックスペース
KEY_F0	ファンクションキー (64 個分の空白が予約されている)
KEY_F(*n*)	For $0 \leq n \leq 63$
KEY_DL	行削除
KEY_IL	行挿入
KEY_DC	文字削除
KEY_IC	文字挿入、または挿入モードに入る
KEY_EIC	挿入モードを抜ける
KEY_CLEAR	画面クリア
KEY_EOS	画面の終わりまでクリア
KEY_EOL	行の終わりまでクリア
KEY_SF	上方へ 1 行スクロール
KEY_SR	下方へ 1 行スクロール (リバース)
KEY_NPAGE	次ページ
KEY_PPAGE	前ページ
KEY_STAB	タブを設定
KEY_CTAB	タブをクリア
KEY_CATAB	全タブをクリア
KEY_ENTER	入力または送信
KEY_SRESET	ソフト (部分的) リセット
KEY_RESET	リセットまたはハードリセット
KEY_PRINT	印刷またはコピー

名前	キー名
KEY_LL	下部のホーム位置 (左下) へ。 キーパッドの割り当ては次のとおり。 A1 up A3 left B2 right C1 down C3
KEY_A1	キーパッドの左上
KEY_A3	キーパッドの右上
KEY_B2	キーパッドの中央
KEY_C1	キーパッドの左下
KEY_C3	キーパッドの右下
KEY_BTAB	バックタブキー
KEY_BEG	開始 (Beginning) キー
KEY_CANCEL	キャンセルキー
KEY_CLOSE	クローズキー
KEY_COMMAND	コマンド (cmd) キー
KEY_COPY	コピーキー
KEY_CREATE	作成キー
KEY_END	終了キー
KEY_EXIT	Exit キー
KEY_FIND	検索キー
KEY_HELP	ヘルプキー
KEY_MARK	マークキー
KEY_MESSAGE	メッセージキー
KEY_MOVE	移動キー
KEY_NEXT	次オブジェクトキー
KEY_OPEN	オープンキー
KEY_OPTIONS	オプションキー
KEY_PREVIOUS	前オブジェクトキー
KEY_REDO	再実行キー
KEY_REFERENCE	参照キー
KEY_REFRESH	リフレッシュキー

名前	キー名
KEY_REPLACE	置換キー
KEY_RESTART	再スタートキー
KEY_RESUME	再開キー
KEY_SAVE	セーブキー
KEY_SBEG	シフト付き開始キー
KEY_SCANCEL	シフト付きキャンセルキー
KEY_SCOMMAND	シフト付きコマンドキー
KEY_SCOPY	シフト付きコピーキー
KEY_SCREATE	シフト付き作成キー
KEY_SDC	シフト付き文字削除キー
KEY_SDL	シフト付き行削除キー
KEY_SELECT	選択キー
KEY_SEND	シフト付き送信キー
KEY_SEOL	シフト付き行クリアキー
KEY_SEXIT	シフト付き Exit キー
KEY_SFIND	シフト付き検索キー
KEY_SHELP	シフト付きヘルプキー
KEY_SHOME	シフト付き Home キー
KEY_SIC	シフト付き入力キー
KEY_SLEFT	シフト付き左向き矢印キー
KEY_SMESSAGE	シフト付きメッセージキー
KEY_SMOVE	シフト付き移動キー
KEY_SNEXT	シフト付き次オブジェクトキー
KEY_SOPTIONS	シフト付きオプションキー
KEY_SPREVIOUS	シフト付き前オブジェクトキー
KEY_SPRINT	シフト付き印刷キー
KEY_SREDO	シフト付き再実行キー
KEY_SREPLACE	シフト付き置換キー
KEY_SRIGHT	シフト付き右向き矢印キー

名前	キー名
KEY_SRSUME	シフト付き再開キー
KEY_SSAVE	シフト付きセーブキー
KEY_SSUSPEND	シフト付き中断キー
KEY_SUNDO	シフト付き取消キー
KEY_SUSPEND	中断キー
KEY_UNDO	取消キー

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目　curses(3CURSES), curs_inopts(3CURSES), curs_move(3CURSES), wrefresh(3CURSES), terminfo(4), attributes(5)

注意事項　ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

単一文字関数に対してエスケープ文字を使うことはできません。

getwch()、wgetwch()、mvgetwch()、mvwgetwch() のいずれかの関数を用いる際、nocbreak モードと echo モードを同時に使用することはできません。個々の文字が入力されたときの tty ドライバの状態によっては、プログラムは予測したものとは異なる結果を生成する場合があります。

getwch()、mvgetwch()、mvwgetwch() の各ルーチンはマクロにすることも可能です。

名前

curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式

cc [*flag...*] *file...* -lcurses [*library...*]

```
#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_addwch, addwch, waddwch, mvaddwch, mvwaddwch, echowchar, wechowchar – wchar_t 文字を属性とともに curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwch(chtype wch);

int waddwch(WINDOW *win, chtype wch);

int mvaddwch(int y, int x, chtype wch);

int mvwaddwch(WINDOW *win, int y, int x, chtype wch);

int echowchar(chtype wch);

int wechowchar(WINDOW *win, chtype wch);
```

機能説明

addwch()、waddwch()、mvaddwch()、mvwaddwch() の各ルーチンは、wchar_t を保ちながら *wch* 文字をウィンドウ中の現カーソル位置に書き出し、ウィンドウカーソルの位置を進めます。この機能は C 複数バイトライブラリ内にある putwchar(3C) の機能に似ています。右側のマージンでは自動的に復帰改行 (newline) が行われます。scrollok が有効であれば、スクロール領域が上に 1 行スクロールされます。

wch がタブ、復帰改行、バックスペースのいずれかの場合、それに従ってウィンドウ内でカーソルが移動します。なお復帰改行の場合には、カーソル移動に先だって clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wch* が他の制御文字の場合、^*X* 形式で描かれます。制御文字を追加した後で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

ビデオ属性は、論理和をパラメータ中にとることにより wchar_t 文字と組み合わせることができます。その結果、属性の設定も行われます。これが意味することは、inwch() と addwch() を使ってテキスト (属性も含む) をある場所から他の場所へコピーすることが可能であるということです。standout(3CURSES) の、定義済みビデオ属性定数の項を参照してください。

機能的に見ると、echowchar() ルーチンは addwch() と refresh(3CURSES) を連続して呼び出すことと同等で、wechowchar() ルーチンは waddwch() と wrefresh(3CURSES) を連続して呼び出すことと同等です。両ルーチンは 1 文字だけしか出力されないことが判っていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。

線グラフィック

addwch() ファミリのルーチンとともに以下の変数を使用して、線を描画する文字を画面に書き出すことができます。端末に変数が定義されているとき、A_ALTCHARSET ビットがオンになります (curs_attr(3CURSES) の項を参照)。変数が定義されていなければ、以下に述べるデフォルト文字が変数に与えられます。なお各変数の名前は VT100 命名規約に準拠しています。

名前	デフォルト	グリフ記述
ACS_ULCORNER	+	左上角
ACS_LLCORNER	+	左下角
ACS_URCORNER	+	右上角
ACS_LRCORNER	+	右下角
ACS_RTEE	+	右端
ACS_LTEE	+	左端
ACS_BTEE	+	下端
ACS_TTEE	+	上端
ACS_HLINE	–	横線
ACS_VLINE	|	縦線
ACS_PLUS	+	正符号
ACS_S1	-	スキャンライン 1
ACS_S9	_	スキャンライン 9
ACS_DIAMOND	+	ダイヤモンド
ACS_CKBOARD	:	市松模様 (点描)
ACS_DEGREE	'	度数記号
ACS_PLMINUS	#	プラス／マイナス
ACS_BULLET	o	丸
ACS_LARROW	<	左向きの矢印
ACS_RARROW	>	右向きの矢印
ACS_DARROW	v	下向きの矢印
ACS_UARROW	^	上向きの矢印
ACS_BOARD	#	正方形のボード
ACS_LANTERN	#	ランタン記号
ACS_BLOCK	#	正方形のブロック

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 putwchar(3C), clrtoeol(3CURSES), curses(3CURSES), curs_attr(3CURSES), curs_inwch(3CURSES), curs_outopts(3CURSES), refresh(3CURSES), standout(3CURSES), winwch(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

addwch()、mvaddwch()、mvwaddwch()、echowchar() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 | curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式 | **cc** [*flag*...] *file*... -lcurses [*library*...]

```
#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明 | 上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

`waddwchnstr()` 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、`chtype` 中でカラー属性を扱うことはできません。

名前 | curs_addwchstr, addwchstr, addwchnstr, waddwchstr, waddwchnstr, mvaddwchstr, mvaddwchnstr, mvwaddwchstr, mvwaddwchnstr – wchar_t 文字列を属性とともに curses ウィンドウに追加

形式 | **cc** [*flag*...] *file*... -lcurses [*library*...]

```
#include<curses.h>

int addwchstr(chtype *wchstr);

int addwchnstr(chtype *wchstr, int n);

int waddwchstr(WINDOW *win, chtype *wchstr);

int waddwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvaddwchstr(int y, int x, chtype *wchstr);

int mvaddwchnstr(int y, int x, chtype *wchstr, int n);

int mvwaddwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwaddwchnstr(WINDOW *win, int y, int x , chtype *wchstr, int n)
    ;
```

機能説明 | 上記ルーチンはいずれも wchar_t 文字列をポイントする *wchstr* を、ウィンドウイメージ構造体中の現カーソル位置に直接コピーします。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個のエレメントをコピーします。ただし行の長さを超えてコピーすることはありません。n=-1 の場合、行に入りきる限り文字列全体を繰り返してコピーします。

ウィンドウカーソルの位置は移動しません。これらのルーチンは、単に *wchstr* をウィンドウイメージ構造体にコピーするだけなので、waddnwstr(3CURSES) よりは処理速度が速くなります。ただし注意すべきことは、復帰改行文字 (newline) があるかなどといったチェックはいっさいしないこと、現カーソル位置を進めないこと、そして行の終わりでは文字列のはみ出した分を (次の行に回さずに) そこで切り捨ててしまうことです。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), waddnwstr(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

`waddwchnstr()` 以外のルーチンはマクロにすることも可能です。 これらのルーチンは、`chtype` 中でカラー属性を扱うことはできません。

名前 | curs_addwstr, addwstr, addnwstr, waddwstr, waddnwstr, mvaddwstr, mvaddnwstr, mvwaddwstr, mvwaddnwstr – wchar_t 文字列を curses ウィンドウに追加してカーソルを進める

形式 | **cc** [*flag...*] *file...* -lcurses [*library...*]

```
#include<curses.h>

int addwstr(wchar_t *wstr);

int addnwstr(wchar_t *wstr, int n);

INT WADDWSTR(WINDOW *WIN, wchar_t *wstr);

int waddnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvaddwstr(int y, int x, wchar_t *wstr);

int mvaddnwstr(int y, int x, wchar_t *wstr, int n);

int mvwaddwstr(WINDOW *win, int y, int x , wchar_t *wstr);

int mvwaddnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | 上記ルーチンはいずれも NULL で終わる wchar_t 文字列 *wstr* を、指定されたウィンドウに書き出します。結果としては文字列中の個々の文字に対して waddwch(3CURSES) を呼び出した場合と同じです。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を書き出します。*n* が負の値の場合、文字列全体が書き出されます。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), waddwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

waddwstr() と waddnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_alecompat, movenextch, wmovenextch, moveprevch, wmoveprevch, adjcurspos, wadjcurspos – カーソルを文字単位に移動するために ALE curses ライブラリに追加された関数

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int movenextch(void);

int wmovenextch(WINDOW *win);

int moveprevch(void);

int wmoveprevch(WINDOW *win);

int adjcurspos(void);

int wadjcurspos(WINDOW *win);
```

機能説明

上記 movenextch() および wmovenextch() ルーチンは、カーソルを次の (右方の) 文字に移動します。次の文字が複数カラム文字であれば、新カーソル位置はその文字の最初の (左端の) カラムとなります。現カーソル位置が複数カラム文字の左端カラムであっても、新しい位置は次の文字上に移動します。なお現カーソル位置が複数カラム文字上の場合、単純なカーソル移動 (++x) を実行しても次の文字に移動するとは保証できません。新しいカーソル位置は getyx(3CURSES) を実行することにより得られます。

moveprevc() および wmoveprevch() ルーチンは、上記 movenextch() と wmovenextch() とは反対方向に、つまり 1 つ前の文字の左端カラムにカーソルを移動します。

adjcurspos() および wadjcurspos() ルーチンは、現在カーソルが置かれている複数カラム文字の左端のカラムへカーソルを移動します。現在の位置がすでに左端カラムである場合、および現在置かれている文字がシングルカラム文字の場合、これらのルーチンは意味を持たず、カーソル位置は変わりません。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getyx(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

movenextch()、moveprevch()、adjcurspos() の各ルーチンはマクロにすることも可能です。

名前

curs_addwch, addwch, waddwch, mvaddwch, mvwaddwch, echowchar, wechowchar – wchar_t 文字を属性とともに curses ウィンドウに追加してカーソルを進める

形式

```
cc [flag...] file... -lcurses [library...]

#include<curses.h>

int addwch(chtype wch);

int waddwch(WINDOW *win, chtype wch);

int mvaddwch(int y, int x, chtype wch);

int mvwaddwch(WINDOW *win, int y, int x, chtype wch);

int echowchar(chtype wch);

int wechowchar(WINDOW *win, chtype wch);
```

機能説明

addwch()、waddwch()、mvaddwch()、mvwaddwch() の各ルーチンは、wchar_t を保ちながら *wch* 文字をウィンドウ中の現カーソル位置に書き出し、ウィンドウカーソルの位置を進めます。この機能は C 複数バイトライブラリ内にある putwchar(3C) の機能に似ています。右側のマージンでは自動的に復帰改行 (newline) が行われます。scrollok が有効であれば、スクロール領域が上に 1 行スクロールされます。

wch がタブ、復帰改行、バックスペースのいずれかの場合、それに従ってウィンドウ内でカーソルが移動します。なお復帰改行の場合には、カーソル移動に先だって clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wch* が他の制御文字の場合、^*X* 形式で描かれます。制御文字を追加した後で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

ビデオ属性は、論理和をパラメータ中にとることにより wchar_t 文字と組み合わせることができます。その結果、属性の設定も行われます。これが意味することは、inwch() と addwch() を使ってテキスト (属性も含む) をある場所から他の場所へコピーすることが可能であるということです。standout(3CURSES) の、定義済みビデオ属性定数の項を参照してください。

機能的に見ると、echowchar() ルーチンは addwch() と refresh(3CURSES) を連続して呼び出すことと同等で、wechowchar() ルーチンは waddwch() と wrefresh(3CURSES) を連続して呼び出すことと同等です。両ルーチンは 1 文字だけしか出力されないことが判っていることを考慮したもので、さらに非制御文字の場合には、2 つのルーチンを連続して呼び出す場合と比較してパフォーマンスの大幅な向上が期待できます。

線グラフィック

addwch() ファミリのルーチンとともに以下の変数を使用して、線を描画する文字を画面に書き出すことができます。端末に変数が定義されているとき、A_ALTCHARSET ビットがオンになります (curs_attr(3CURSES) の項を参照)。変数が定義されていなければ、以下に述べるデフォルト文字が変数に与えられます。なお各変数の名前は VT100 命名規約に準拠しています。

名前	デフォルト	グリフ記述
ACS_ULCORNER	+	左上角
ACS_LLCORNER	+	左下角
ACS_URCORNER	+	右上角
ACS_LRCORNER	+	右下角
ACS_RTEE	+	右端
ACS_LTEE	+	左端
ACS_BTEE	+	下端
ACS_TTEE	+	上端
ACS_HLINE	–	横線
ACS_VLINE	|	縦線
ACS_PLUS	+	正符号
ACS_S1	-	スキャンライン 1
ACS_S9	_	スキャンライン 9
ACS_DIAMOND	+	ダイヤモンド
ACS_CKBOARD	:	市松模様 (点描)
ACS_DEGREE	'	度数記号
ACS_PLMINUS	#	プラス／マイナス
ACS_BULLET	o	丸
ACS_LARROW	<	左向きの矢印
ACS_RARROW	>	右向きの矢印
ACS_DARROW	v	下向きの矢印
ACS_UARROW	^	上向きの矢印
ACS_BOARD	#	正方形のボード
ACS_LANTERN	#	ランタン記号
ACS_BLOCK	#	正方形のブロック

戻り値　上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します (それ以外の動作が本項内で記述されている場合を除く)。

属性　次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 putwchar(3C), clrtoeol(3CURSES), curses(3CURSES), curs_attr(3CURSES), curs_inwch(3CURSES), curs_outopts(3CURSES), refresh(3CURSES), standout(3CURSES), winwch(3CURSES), wrefresh(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

addwch()、mvaddwch()、mvwaddwch()、echowchar() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_getwch, getwch, wgetwch, mvgetwch, mvwgetwch, ungetwch – curses 端末キーボードから wchar_t 文字を得る (または戻す)

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwch(void);

int wgetwch(WINDOW *win);

int mvgetwch(int y, int x);

int mvwgetwch(WINDOW *win, int y, int x);

int ungetwch(int wch);
```

機能説明

getwch()、wgetwch()、mvgetwch()、mvwgetwch() の各ルーチンは、ウィンドウに対応づけられた端末から EUC 文字 1 文字を読み込み、それを wchar_t 文字に転送し、wchar_t 文字を返します。遅延無しモードでは、入力待ちのものがなければ ERR 値が返されます。遅延モードでは、システムからテキストが渡されるまでプログラムは待ち状態に置かれます。待つタイミングは cbreak の設定により異なり、1 つの文字の後 (cbreak モード) または最初の改行の後 (nocbreak モード) となります。また半遅延モードでは、何らかの文字が入力されるまであるいは指定されたタイムアウト値に達するまで待ちます。noecho が設定されていなければ、読み込まれた文字は指定されたウィンドウ中にエコーされます。

ウィンドウがパッドでなく、最後に wrefresh(3CURSES) が呼び出されてからウィンドウの移動または変更が行われていれば、次の文字を読み込む前に wrefresh が呼び出されます。

keypad が TRUE の場合、ファンクションキーが押されると実際の文字の代わりにファンクションキーのトークンが返されます。使用可能なファンクションキーは KEY_ で始まる名前で、0401 で始まる整数とともに <curses.h> 中に定義されています。ファンクションキーの先頭文字でありうる文字 (たとえばエスケープ文字) を受け取ると、curses(3CURSES) はタイマをセットします。指定された時間内にそのシーケンスの残りの文字が到着しなければ、受け取った 1 文字だけを渡します。時間内に残りの文字を受け取れば、ファンクションキーの値を返します。このため、エスケープ文字を押したのにそれがプログラムに渡されるまでに時間がかかった、 というケースを多くの端末が経験することになります。

ungetwch() ルーチンは、次に wgetwch() を呼び出したときに返される入力キュー中に *wch* 文字を戻します。

ファンクションキー

keypad が有効であれば、以下の表にリストされているファンクションキー (いずれも <curses.h> 中に定義) が getwch() により返されます。なお端末によっては、これらすべてのファンクションキーがサポートされるとは限りません。サポートされないケースは、そのキーが押されたときにその端末転送するコードが一意ではない場合、およびそのキーの定義が terminfo(4) データベース中に存在しない場合です。

名前	キー名
KEY_BREAK	Break キー
KEY_DOWN	4 つの矢印キー ...
KEY_UP	
KEY_LEFT	
KEY_RIGHT	
KEY_HOME	Home キー (左上向き矢印)
KEY_BACKSPACE	バックスペース
KEY_F0	ファンクションキー (64 個分の空白が予約されている)
KEY_F(*n*)	For $0 \leq n \leq 63$
KEY_DL	行削除
KEY_IL	行挿入
KEY_DC	文字削除
KEY_IC	文字挿入、または挿入モードに入る
KEY_EIC	挿入モードを抜ける
KEY_CLEAR	画面クリア
KEY_EOS	画面の終わりまでクリア
KEY_EOL	行の終わりまでクリア
KEY_SF	上方へ 1 行スクロール
KEY_SR	下方へ 1 行スクロール (リバース)
KEY_NPAGE	次ページ
KEY_PPAGE	前ページ
KEY_STAB	タブを設定
KEY_CTAB	タブをクリア
KEY_CATAB	全タブをクリア
KEY_ENTER	入力または送信
KEY_SRESET	ソフト (部分的) リセット
KEY_RESET	リセットまたはハードリセット
KEY_PRINT	印刷またはコピー

名前	キー名
KEY_LL	下部のホーム位置 (左下) へ。 キーパッドの割り当ては次のとおり。 A1 up A3 left B2 right C1 down C3
KEY_A1	キーパッドの左上
KEY_A3	キーパッドの右上
KEY_B2	キーパッドの中央
KEY_C1	キーパッドの左下
KEY_C3	キーパッドの右下
KEY_BTAB	バックタブキー
KEY_BEG	開始 (Beginning) キー
KEY_CANCEL	キャンセルキー
KEY_CLOSE	クローズキー
KEY_COMMAND	コマンド (cmd) キー
KEY_COPY	コピーキー
KEY_CREATE	作成キー
KEY_END	終了キー
KEY_EXIT	Exit キー
KEY_FIND	検索キー
KEY_HELP	ヘルプキー
KEY_MARK	マークキー
KEY_MESSAGE	メッセージキー
KEY_MOVE	移動キー
KEY_NEXT	次オブジェクトキー
KEY_OPEN	オープンキー
KEY_OPTIONS	オプションキー
KEY_PREVIOUS	前オブジェクトキー
KEY_REDO	再実行キー
KEY_REFERENCE	参照キー
KEY_REFRESH	リフレッシュキー

名前	キー名
KEY_REPLACE	置換キー
KEY_RESTART	再スタートキー
KEY_RESUME	再開キー
KEY_SAVE	セーブキー
KEY_SBEG	シフト付き開始キー
KEY_SCANCEL	シフト付きキャンセルキー
KEY_SCOMMAND	シフト付きコマンドキー
KEY_SCOPY	シフト付きコピーキー
KEY_SCREATE	シフト付き作成キー
KEY_SDC	シフト付き文字削除キー
KEY_SDL	シフト付き行削除キー
KEY_SELECT	選択キー
KEY_SEND	シフト付き送信キー
KEY_SEOL	シフト付き行クリアキー
KEY_SEXIT	シフト付き Exit キー
KEY_SFIND	シフト付き検索キー
KEY_SHELP	シフト付きヘルプキー
KEY_SHOME	シフト付き Home キー
KEY_SIC	シフト付き入力キー
KEY_SLEFT	シフト付き左向き矢印キー
KEY_SMESSAGE	シフト付きメッセージキー
KEY_SMOVE	シフト付き移動キー
KEY_SNEXT	シフト付き次オブジェクトキー
KEY_SOPTIONS	シフト付きオプションキー
KEY_SPREVIOUS	シフト付き前オブジェクトキー
KEY_SPRINT	シフト付き印刷キー
KEY_SREDO	シフト付き再実行キー
KEY_SREPLACE	シフト付き置換キー
KEY_SRIGHT	シフト付き右向き矢印キー

名前	キー名
KEY_SRSUME	シフト付き再開キー
KEY_SSAVE	シフト付きセーブキー
KEY_SSUSPEND	シフト付き中断キー
KEY_SUNDO	シフト付き取消キー
KEY_SUSPEND	中断キー
KEY_UNDO	取消キー

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inopts(3CURSES), curs_move(3CURSES), wrefresh(3CURSES), terminfo(4), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

単一文字関数に対してエスケープ文字を使うことはできません。

getwch()、wgetwch()、mvgetwch()、mvwgetwch() のいずれかの関数を用いる際、nocbreak モードと echo モードを同時に使用することはできません。個々の文字が入力されたときの tty ドライバの状態によっては、プログラムは予測したものとは異なる結果を生成する場合があります。

getwch()、mvgetwch()、mvwgetwch() の各ルーチンはマクロにすることも可能です。

名前

curs_getwstr, getwstr, getnwstr, wgetwstr, wgetnwstr, mvgetwstr, mvgetnwstr, mvwgetwstr, mvwgetnwstr – curses 端末キーボードから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int getwstr(wchar_t *wstr);

int getnwstr(wchar_t *wstr, int n);

int wgetwstr(WINDOW *win, wchar_t *wstr);

int wgetnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvgetwstr(int y, int x, wchar_t *wstr);

int mvgetnwstr(int y, int x, wchar_t *wstr, int n);

int mvwgetwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwgetnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

getwstr() の実行結果は、復帰改行とキャリッジリターンを受け取るまで getwch(3CURSES) を連続して呼び出した場合の結果と同等です。実行結果は、wchar_t ポインタ *wstr* が示す領域に置かれます。getnwstr() は最大 *n* 個の wchar_t 文字を読み込みます。この最大文字数制限により、入力バッファがオーバフローしないようにします。ユーザーの消去 (erase) 文字や削除 (kill) 文字、およびファンクションキー、HOME キー、CLEAR キーなどの特殊キーも解釈されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

wgetnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明 | これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の `wchar_t` 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の `wchar_t` 文字を返します。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 `ERR` を返し、正常に終了すれば `ERR` 以外の整数値を返します。

属性 | 次の属性については `attributes`(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | `curses`(3CURSES), `attributes`(5)

注意事項 | ヘッダーファイル `<curses.h>` は自動的に `<stdio.h>`、`<unctrl.h>`、および `<widec.h>` ヘッダーファイルを含みます。

`winnwstr()` 以外のルーチンはマクロにすることも可能です。

名前

curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^X 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 | curs_inswch, inswch, winswch, mvinswch, mvwinswch – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswch(chtype wch);

int winswch(WINDOW *win, chtype wch);

int mvinswch(int y, int x, chtype wch);

int mvwinswch(WINDOW *win, int y, int x, chtype wch);
```

機能説明 | これらのルーチンは、wchar_t 文字を含んでいる *wch* を、現カーソル位置の文字の直前に挿入します。これによりカーソルの右側にある文字はそれぞれ 1 文字分ずつ右に移動し、その結果、行の右端にある文字が失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用する ということを意味するものではありません。

戻り値 | 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 | 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 | curses(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

inswch()、mvinswch()、mvwinswch() の各ルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inswstr, inswstr, insnwstr, winswstr, winsnwstr, mvinswstr, mvinsnwstr, mvwinswstr, mvwinsnwstr – curses ウィンドウ内のカーソル位置の文字の直前に wchar_t 文字列を挿入

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inswstr(wchar_t *wstr);

int insnwstr(wchar_t *wstr, int n);

int winswstr(WINDOW *win, wchar_t *wstr);

int winsnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinswstr(int y, int x, wchar_t *wstr);

int mvinsnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinswstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinsnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、wchar_t 文字列を現カーソル位置の文字の直前に挿入します。挿入可能な最大文字数は、その行に入りきる数です。挿入によりカーソルの右側にある文字は右に移動し、その結果行の右端の何文字かが失われる可能性もあります。カーソル位置は (*y*、*x* が指定されていればその位置に移動後) 変更されません。なお、これはハードウェアの文字挿入機構を使用するということを意味するものではありません。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字をコピーします。*n*<=0 の場合、文字列全体をコピーします。

wstr 中にタブ、復帰改行 (newline)、キャリッジリターン、またはバックスペース文字が含まれていると、カーソルはウィンドウ内で移動してその文字で指定された位置に置かれます。なお復帰改行の場合には、カーソル移動に先立って clrtoeol(3CURSES) が行われます。タブ位置は 8 カラムごとに設定されていると見なされます。*wstr* 中に他の制御文字がある場合、^*X* 形式で描かれます。制御文字を追加した後 (さらに必要ならばカーソルをその位置に移動した後) で winwch(3CURSES) を呼び出しても制御文字自体は返されません。その代わりに制御文字の表示内容が返されます。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

clrtoeol(3CURSES), curses(3CURSES), winwch(3CURSES), attributes(5)

注意事項 | ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。 winsnwstr() 以外のルーチンはマクロにすることも可能です。

名前 curs_inwch, inwch, winwch, mvinwch, mvwinwch – curses ウィンドウから wchar_t 文字とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

chtype inwch(void);

chtype winwch(WINDOW *win);

chtype mvinwch(int y, int x);

chtype mvwinwch(WINDOW *win, int y, int x);
```

機能説明 これらのルーチンは、指定されたウィンドウ中の現カーソル位置にある、chtype タイプの wchar_t 文字を返します。その位置に属性が設定されていれば、それらの属性値の論理和も返されます。<curses.h> 中に定義されている定数を & (論理積) 演算子とともに用いて、文字だけもしくは属性だけを抽出することも可能です。

属性 winwch() が返す値と以下に示すビットマスクとの論理積をとることもできます。

A_WCHARTEXT	文字抽出用のビットマスク
A_WATTRIBUTES	属性抽出用のビットマスク

属性 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 curses(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

全ルーチンともマクロにすることが可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前 curs_inwchstr, inwchstr, inwchnstr, winwchstr, winwchnstr, mvinwchstr, mvinwchnstr , mvwinwchstr, mvwinwchnstr – curses ウィンドウから wchar_t 文字列とその属性を得る

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int inwchstr(chtype *wchstr);

int inwchnstr(chtype *wchstr, int n);

int winwchstr(WINDOW *win, chtype *wchstr);

int winwchnstr(WINDOW *win, chtype *wchstr, int n);

int mvinwchstr(int y, int x, chtype *wchstr);

int mvinwchnstr(int y, int x, chtype *wchstr, int n);

int mvwinwchstr(WINDOW *win, int y, int x, chtype *wchstr);

int mvwinwchnstr(WINDOW *win, int y, int x, chtype *wchstr, int n);
```

機能説明 これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる、wchar_t 文字からなる chtype タイプの文字列を返します。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。<curses.h> 中に定義されている定数を論理積 (&) 演算子とともに用いて、*wchstr* 中の任意の位置から wchar_t 文字だけもしくは属性だけを抽出することも可能です (詳しくは curs_inwch(3CURSES) の説明を参照してください)。

戻り値 上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性 次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目 curses(3CURSES), curs_inwch(3CURSES), attributes(5)

注意事項 ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winwchnstr() 以外のルーチンはマクロにすることも可能です。

これらのルーチンは、chtype 中でカラー属性を扱うことはできません。

名前

curs_inwstr, inwstr, innwstr, winwstr, winnwstr, mvinwstr, mvinnwstr, mvwinwstr, mvwinnwstr – curses ウィンドウから wchar_t 文字列を得る

形式

```
cc [ flag ... ] file ... -lcurses[library .. ]
#include <curses.h>

int inwstr(wchar_t *wstr);

int innwstr(wchar_t *wstr, int n);

int winwstr(WINDOW *win, wchar_t *wstr);

int winnwstr(WINDOW *win, wchar_t *wstr, int n);

int mvinwstr(int y, int x, wchar_t *wstr);

int mvinnwstr(int y, int x, wchar_t *wstr, int n);

int mvwinwstr(WINDOW *win, int y, int x, wchar_t *wstr);

int mvwinnwstr(WINDOW *win, int y, int x, wchar_t *wstr, int n);
```

機能説明

これらのルーチンは、指定されたウィンドウ中の現カーソル位置から始まり右マージンで終わる *wstr* 中の wchar_t 文字列を返します。属性値は捨てられます。最終引数として *n* を指定する 4 つのルーチンは、最大 *n* 個の wchar_t 文字を返します。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

winnwstr() 以外のルーチンはマクロにすることも可能です。

名前

curs_alecompat, movenextch, wmovenextch, moveprevch, wmoveprevch, adjcurspos, wadjcurspos – カーソルを文字単位に移動するために ALE curses ライブラリに追加された関数

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int movenextch(void);

int wmovenextch(WINDOW *win);

int moveprevch(void);

int wmoveprevch(WINDOW *win);

int adjcurspos(void);

int wadjcurspos(WINDOW *win);
```

機能説明

上記 movenextch() および wmovenextch() ルーチンは、カーソルを次の (右方の) 文字に移動します。次の文字が複数カラム文字であれば、新カーソル位置はその文字の最初の (左端の) カラムとなります。現カーソル位置が複数カラム文字の左端カラムであっても、新しい位置は次の文字上に移動します。なお現カーソル位置が複数カラム文字上の場合、単純なカーソル移動 (++x) を実行しても次の文字に移動するとは保証できません。新しいカーソル位置は getyx(3CURSES) を実行することにより得られます。

moveprevc() および wmoveprevch() ルーチンは、上記 movenextch() と wmovenextch() とは反対方向に、つまり 1 つ前の文字の左端カラムにカーソルを移動します。

adjcurspos() および wadjcurspos() ルーチンは、現在カーソルが置かれている複数カラム文字の左端のカラムへカーソルを移動します。現在の位置がすでに左端カラムである場合、および現在置かれている文字がシングルカラム文字の場合、これらのルーチンは意味を持たず、カーソル位置は変わりません。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getyx(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

movenextch()、moveprevch()、adjcurspos() の各ルーチンはマクロにすることも可能です。

名前

curs_alecompat, movenextch, wmovenextch, moveprevch, wmoveprevch, adjcurspos, wadjcurspos – カーソルを文字単位に移動するために ALE curses ライブラリに追加された関数

形式

```
cc [ flag ... ] file ... -lcurses [ library .. ]
#include <curses.h>

int movenextch(void);

int wmovenextch(WINDOW *win);

int moveprevch(void);

int wmoveprevch(WINDOW *win);

int adjcurspos(void);

int wadjcurspos(WINDOW *win);
```

機能説明

上記 movenextch() および wmovenextch() ルーチンは、カーソルを次の (右方の) 文字に移動します。次の文字が複数カラム文字であれば、新カーソル位置はその文字の最初の (左端の) カラムとなります。現カーソル位置が複数カラム文字の左端カラムであっても、新しい位置は次の文字上に移動します。なお現カーソル位置が複数カラム文字上の場合、単純なカーソル移動 (++x) を実行しても次の文字に移動するとは保証できません。新しいカーソル位置は getyx(3CURSES) を実行することにより得られます。

moveprevc() および wmoveprevch() ルーチンは、上記 movenextch() と wmovenextch() とは反対方向に、つまり 1 つ前の文字の左端カラムにカーソルを移動します。

adjcurspos() および wadjcurspos() ルーチンは、現在カーソルが置かれている複数カラム文字の左端のカラムへカーソルを移動します。現在の位置がすでに左端カラムである場合、および現在置かれている文字がシングルカラム文字の場合、これらのルーチンは意味を持たず、カーソル位置は変わりません。

戻り値

上記ルーチンはすべて、エラーが発生すれば整数 ERR を返し、正常に終了すれば ERR 以外の整数値を返します。

属性

次の属性については attributes(5) のマニュアルページを参照してください。

属性タイプ	属性値
MT レベル	Unsafe

関連項目

curses(3CURSES), getyx(3CURSES), attributes(5)

注意事項

ヘッダーファイル <curses.h> は自動的に <stdio.h>、<unctrl.h>、および <widec.h> ヘッダーファイルを含みます。

movenextch()、moveprevch()、adjcurspos() の各ルーチンはマクロにすることも可能です。