



SunOS リファレンスマニュアル 3 : ライブラリインタフェースおよび ヘッダー

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

Part No: 817-5084-10
2004 年 4 月

Copyright 2004 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

本製品およびそれに関連する文書は著作権法により保護されており、その使用、複製、頒布および逆コンパイルを制限するライセンスのもとにおいて頒布されます。サン・マイクロシステムズ株式会社による事前の許可なく、本製品および関連する文書のいかなる部分も、いかなる方法によっても複製することが禁じられます。

本製品の一部は、カリフォルニア大学からライセンスされている Berkeley BSD システムに基づいていることがあります。UNIX は、X/Open Company, Ltd. が独占的にライセンスしている米国ならびに他の国における登録商標です。フォント技術を含む第三者のソフトウェアは、著作権により保護されており、提供者からライセンスを受けているものです。

Federal Acquisitions: Commercial Software—Government Users Subject to Standard License Terms and Conditions.

本製品に含まれる HG-MinchoL、HG-MinchoL-Sun、HG-PMinchoL-Sun、HG-GothicB、HG-GothicB-Sun、および HG-PGothicB-Sun は、株式会社リコーがリコービマジクス株式会社からライセンス供与されたタイプフェイスマスタをもとに作成されたものです。HeiseiMin-W3H は、株式会社リコーが財団法人日本規格協会からライセンス供与されたタイプフェイスマスタをもとに作成されたものです。フォントとして無断複製することは禁止されています。

Sun、Sun Microsystems、docs.sun.com、AnswerBook、AnswerBook2、JumpStart、Solaris Web Start、Power Management、Sun ONE Application Server、Solaris Flash、Solaris Live Upgrade は、米国およびその他の国における米国 Sun Microsystems, Inc. (以下、米国 Sun Microsystems 社とします) の商標もしくは登録商標です。

サンのロゴマークおよび Solaris は、米国 Sun Microsystems 社の登録商標です。

すべての SPARC 商標は、米国 SPARC International, Inc. のライセンスを受けて使用している同社の米国およびその他の国における商標または登録商標です。SPARC 商標が付いた製品は、米国 Sun Microsystems 社が開発したアーキテクチャに基づくものです。

OPENLOOK、OpenBoot、JLE は、サン・マイクロシステムズ株式会社の登録商標です。

Wnn は、京都大学、株式会社アステック、オムロン株式会社で共同開発されたソフトウェアです。

Wnn6 は、オムロン株式会社、オムロンソフトウェア株式会社で共同開発されたソフトウェアです。© Copyright OMRON Co., Ltd. 1995-2000. All Rights Reserved. © Copyright OMRON SOFTWARE Co., Ltd. 1995-2002 All Rights Reserved.

「ATOK」は、株式会社ジャストシステムの登録商標です。

「ATOK Server/ATOK12」は、株式会社ジャストシステムの著作物であり、「ATOK Server/ATOK12」にかかる著作権その他の権利は、株式会社ジャストシステムおよび各権利者に帰属します。

本製品に含まれる郵便番号辞書 (7 桁/5 桁) は郵政事業庁が公開したデータを元に制作された物です (一部データの加工を行なっています)。

本製品に含まれるフェイスマーク辞書は、株式会社ビレッジセンターの許諾のもと、同社が発行する『インターネット・パソコン通信フェイスマークガイド '98』に添付のものを使用しています。© 1997 ビレッジセンター

Unicode は、Unicode, Inc. の商標です。

本書で参照されている製品やサービスに関しては、該当する会社または組織に直接お問い合わせください。

OPEN LOOK および Sun Graphical User Interface は、米国 Sun Microsystems 社が自社のユーザおよびライセンス実施権者向けに開発しました。米国 Sun Microsystems 社は、コンピュータ産業用のビジュアルまたはグラフィカル・ユーザインタフェースの概念の研究開発における米国 Xerox 社の先駆者としての成果を認めるものです。米国 Sun Microsystems 社は米国 Xerox 社から Xerox Graphical User Interface の非独占的ライセンスを取得しており、このライセンスは米国 Sun Microsystems 社のライセンス実施権者にも適用されます。

DtComboBox ウィジェットと DtSpinBox ウィジェットのプログラムおよびドキュメントは、Interleaf, Inc. から提供されたものです。(© 1993 Interleaf, Inc.)

本書は、「現状のまま」をベースとして提供され、商品性、特定目的への適合性または第三者の権利の非侵害の黙示の保証を含みそれに限定されず、明示的であるか黙示的であるかを問わない、なんらの保証も行われぬものとします。

本製品が、外国為替および外国貿易管理法 (外為法) に定められる戦略物資等 (貨物または役務) に該当する場合、本製品を輸出または日本国外へ持ち出す際には、サン・マイクロシステムズ株式会社の事前の書面による承諾を得ることのほか、外為法および関連法規に基づく輸出手続き、また場合によっては、米国商務省または米国所轄官庁の許可を得ることが必要です。

原典: *man pages section 3: Library Interfaces and Headers*

Part No: 817-0713-10

Revision A



040304@7940



目次

はじめに 5

SunOS リファレンスマニュアル 3: ライブラリインタフェースおよびヘッダー 9

Intro(3) 10

libcfgadm(3LIB) 32

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 4 は、各種ファイルの形式について説明します。また、ファイル形式を宣言する C 構造体を適用できる場合には随時説明しています。
- 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。
- セクション 7 は、特殊なハードウェア周辺装置またはデバイスドライバに関するさまざまな特殊ファイルについて説明します。STREAMS ソフトウェアドライバ、モジュール、またはシステムコールの STREAMS 汎用セットについても説明します。

- セクション9は、カーネル環境でデバイスドライバを記述するのに必要な参照情報を提供します。ここでは、デバイスドライバインタフェース (DDI) とドライバ/カーネルインタフェース (DKI) という2つのデバイスドライバインタフェース仕様について説明します。
- セクション9Fは、デバイスドライバが使用できるカーネル関数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	<p>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。</p> <p>以下の文字がそれぞれの項目で使われています。</p> <p>[] 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。</p> <p>... 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...')。</p> <p> 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。</p> <p>{ } 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。</p>
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

	<p>デバイスに関する <code>ioctl</code> 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 <code>ioctl</code> は、<code>mtio(7I)</code> のように <code>io</code> 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。</p>
オプション	<p>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。</p>
オペランド	<p>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。</p>
出力	<p>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。</p>
戻り値	<p>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。</p>
エラー	<p>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 <code>errno</code> 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。</p>
使用法	<p>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。</p>
	<p>コマンド 修飾子 変数 式 入力文法</p>
使用例	<p>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 <code>example%</code> のプロンプトが出てきます。スーパーユーザーの場合は <code>example#</code> 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。</p>
環境	<p>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。</p>

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せん。詳細は <code>attributes(5)</code> 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 (<i>Italic</i>) 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: ラ イブラリインタフェースおよび ヘッダー

Intro(3)

名前	Intro, intro – 関数とライブラリの紹介
機能説明	<p>このセクションでは、さまざまな Solaris ライブラリにある関数のうち、直接 UNIX システムのプリミティブを呼び出す関数 (このマニュアルページのセクション 2 で説明) 以外の関数について説明します。関数の宣言は、各ページで説明されている <code>#include</code> ファイルから取得できます。このセクションの説明はライブラリ別になっています。各ライブラリは、セクション番号の後ろに入っているライブラリ名 (ライブラリの省略名) で識別できます。関連のあるライブラリのコレクションは、下記に示すように大きく 5 つのボリュームに分かれており、6 番目のボリュームは、この 5 つのボリュームで説明されている関数、マクロ、外部変数が使用する共有ライブラリとヘッダーの内容説明になっています。ここでは、まず、この 6 番目のボリュームから説明します。</p>
ライブラリインタフェースおよびヘッダー	<p>このボリュームでは、他の 5 つのボリュームで説明されている関数、マクロ、外部変数が使用する共有ライブラリとヘッダーの内容について説明します。</p> <p>(3LIB) このセクションで説明するライブラリはどれも、共有オブジェクトとして実装されています。</p> <p>共有オブジェクトの説明には、共有オブジェクトの共有インタフェースを定義する大域シンボルの定義が含まれていることがあります (例: <code>SUNW_1.1</code>)。このほかのインタフェースは、共有オブジェクトに含まれている可能性があります (例: <code>SUNW_private.1.1</code>)。共有インタフェースは、アプリケーション開発用に、安定度の高い専用のシンボルセットを提供します。専用インタフェースは内部使用のみであり、随時変更される可能性があります。</p> <p>共有オブジェクトの多くには、32 ビットのシステムのみで下位互換性を提供するアーカイブライブラリが用意されています。このようなライブラリを使用すると、アプリケーションによっては別の Solaris リリースに移行することができなくなる場合があります。このため、Solaris では、このようなライブラリの代わりにダイナミックリンクを使用するコンパイル方式が優先されます。</p> <p>(3LIBUCB) このセクションで説明する SunOS/BSD 互換ライブラリは、共有オブジェクトとして実装されています。上記の (3LIB) を参照してください。</p> <p>(3HEAD) このセクションで説明するヘッダーは、関数、マクロ、外部変数が使用するヘッダーです。ヘッダーには、関数のプロトタイプ、記号定数の定義、共通の構造体、プリプロセッサマクロ、定義済みの型が含まれています。残りの 5 つのボリュームで説明されている各関数は、この関数を使用するためにアプリケーションに含めなければならないヘッダーを指定します。多くの場合、必要なヘッダーは 1 つだけです。このようなヘッダーは、アプリケーション開発システムに配置されます。実行システムに配置する必要はありません。</p>

基本ライブラリ関数

このボリュームで説明する関数は、アプリケーション開発の基本となるコア C ライブラリ関数です。

- (3C) セクション 2 の関数とともに標準 C ライブラリ `libc` を構成する関数です。`libc` は、C コンパイルシステムにより自動的にリンクされます。標準 C ライブラリは共有オブジェクト `libc.so`、アーカイブ `libc.a` として実装されています。C プログラムは、デフォルトで、標準 C ライブラリの共有オブジェクトバージョンにリンクされます。アーカイブバージョンにリンクするには、`cc` コマンド行で `-Bstatic` または `-dn` を指定します。このほかの優先指定については、`libc(3LIB)`、`cc(1B)` を参照してください。
『ANSI C Programmer's Guide』の「C Compilation System」の章にも参考になる情報があります。規格に合致した環境では、動作が異なる関数もあります。こうした動作については、個々のマニュアルページで説明されています。`standards(5)` を参照してください。
- (3DL) ダイナミックリンクライブラリ `libdl` を構成する関数です。このライブラリは共有オブジェクト `libdl.so` として実装されていますが、C コンパイルシステムによって自動的にリンクされません。`cc` コマンド行で `-ldl` を指定すると、このライブラリにリンクできます。`libdl(3LIB)` を参照してください。
- (3MALLOC) `libmalloc`、`libbsdmalloc`、`libmapmalloc`、`libmtmalloc` という複数のメモリ割り当てライブラリを構成する関数です。これらのライブラリはそれぞれ共有オブジェクト `libmalloc.so`、`libbsdmalloc.so`、`libmapmalloc.so`、`libmtmalloc.so` として実装されています。このうち `libmtmalloc` 以外のすべては、アーカイブ `libmalloc.a`、`libbsdmalloc.a`、`libmapmalloc.a` としても実装されています。このライブラリ群は、C コンパイルシステムによって自動的にリンクされません。`libmalloc`、`libbsdmalloc`、`libmapmalloc`、`libmtmalloc` とリンクするには、それぞれ `-lmalloc`、`-lbsdmalloc`、`-lmapmalloc`、`-lmtmalloc` の各オプションを指定します。`libmalloc(3LIB)`、`libbsdmalloc(3LIB)`、`libmapmalloc(3LIB)`、`libmtmalloc(3LIB)` を参照してください。
- (3UCB) BSD 関数を含むソース互換ライブラリを構成する関数です。共有オブジェクト `libucb.so`、アーカイブ `libucb.a` として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリは `/usr/ucb` サブディレクトリにあるため、`cc` コマンド行で `-lucb` を指定することによってリンクできます。このライブラリのヘッダーは、`/usr/ucbinclude` に入っています。`libucb(3LIBUCB)` を参照してください。

ネットワークライブラリ関数

このボリュームで説明する関数は、さまざまなネットワークライブラリを構成します。

Intro(3)

- (3GSS) このライブラリ中の関数は、Generic Security Services API (GSS-API) ライブラリを構成するルーチンです。このライブラリは共有オブジェクト `libgss.so.1` として実装されています。ただし、C コンパイルシステムによって自動的にはリンクされません。このライブラリにリンクするには、`cc` コマンド行で `-lgss` を指定します。`libgss(3LIB)` を参照してください。
- (3LDAP) 軽量ディレクトリアクセスプロトコルライブラリを構成する関数です。このライブラリは共有オブジェクト `libldap.so` として実装されていますが、C コンパイルシステムによって自動的にはリンクされません。このライブラリにリンクするには、`cc` コマンド行で `-lldap` を指定します。`ldap(3LDAP)` を参照してください。
- (3NSL) ネットワークサービ斯拉イブラリ `libnsl` を構成する関数です。このライブラリは共有オブジェクト `libnsl.so`、アーカイブ `libnsl.a` として実装されていますが、C コンパイルシステムによって自動的にはリンクされません。このライブラリにリンクするには、`cc` コマンド行で `-lnsl` を指定します。`libnsl(3LIB)` を参照してください。
- 基本ネットワーキング関数の多くは、X/Open ネットワーキングインタフェースライブラリにも入っています。`libxnet` インタフェースの詳細については、下記のセクション (`3XNET`) を参照してください。
- (3RAC) リモート非同期呼び出しライブラリ `librac` を構成する関数です。このライブラリは共有オブジェクト `librac.so`、アーカイブ `librac.a` として実装されていますが、C コンパイルシステムによって自動的にはリンクされません。このライブラリにリンクするには、`cc` コマンド行で `-lrac` を指定します。`librac(3LIB)` を参照してください。
- (3RESOLV) リゾルバライブラリ `libresolv` を構成する関数です。このライブラリは共有オブジェクト `libresolv.so`、アーカイブ `libresolv.a` として実装されていますが、C コンパイルシステムによって自動的にはリンクされません。このライブラリにリンクするには、`cc` コマンド行で `-lresolv` を指定します。`libresolv(3LIB)` を参照してください。
- (3RPC) 遠隔手続き呼び出しライブラリ `librpcsvc` と `librpcsoc` を構成する関数です。`librpcsoc` は互換性の提供のみを目的としたライブラリなので、これには新しいアプリケーションをリンクしないでください。この2つのライブラリは、それぞれ共有オブジェクト `librpcsvc.so` と `librpcsoc.so` として、また、アーカイブ `librpcsvc.a` として実装されています (`librt(3LIB)` 参照)。どちらのライブラリも、C コンパイルシステムによって自動的にはリンクされません。これらにリンクするには、`cc` コマンド行で `-lrpcsvc` か `-lrpcsoc` を指定します。`librpcsvc(3LIB)` および `librpcsoc(3LIBUCB)` を参照してください。

- (3SLP) SLP (サービスロケーションプロトコル, Service Location Protocol) ライブラリ `libslp` を構成する関数です。このライブラリは、共有オブジェクト `libslp.so.1` として実装されていますが、C コンパイルシステムによって自動的にリンクされません。 `libslp(3LIB)` を参照してください。
- (3SOCKET) ソケットライブラリ `libsocket` を構成する関数です。このライブラリは、共有オブジェクト `libsocket.so`、アーカイブ `libsocket.a` として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で `-lsocket` を指定します。 `libsocket(3LIB)` を参照してください。
- (3XFN) X/Open フェデレーテッドネーミングライブラリ `libxfn` を構成する関数です。このライブラリは共有オブジェクト `libxfn.so` として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で `-lxfn` を指定します。 `libxfn(3LIB)`、`xfn(3XFN)`、`fns(5)`、`standards(5)` を参照してください。
- (3XNET) X/Open CAE 仕様『Networking Services』(Issue 4、1994年9月)に準拠した X/Open ネットワーキングインタフェースを構成する関数です。このライブラリは共有オブジェクト `libxnet.so` として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で `-lxnet` を指定します。コンパイル情報については、`libxnet(3LIB)` および `standards(5)` を参照してください。

どのような状況でも、XTI API や TLI API ではなく、ソケット API を優先的に使用することをお勧めします。ただし、ほかの XPGV4v2 (`standards(5)` を参照) システムに移植する必要がある場合は、アプリケーションで `libxnet` インタフェースを使用する必要があります。移植の必要がない場合は、`libxnet` のインタフェースではなく、`libsocket` や `libnsl` のソケットインタフェースを使用する方法をお勧めします。XTI API、TLI API の中では、`libnsl` で使用できる TLI インタフェースよりも `libxnet` で使用できる XTI インタフェースのほうをお勧めします。

Curses ライブラリ関数

このボリュームで説明する関数は、グラフィックスと文字が表示された画面の更新機能を提供するライブラリを構成します。

- (3CURSES) 次のライブラリを構成する関数です。
- | | |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>libcurses</code> | <code>curses</code> ライブラリ <code>libcurses</code> を構成する関数です。このライブラリは共有オブジェクト <code>libcurses.so</code> 、アーカイブ <code>libcurses.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-lcurses</code> を指定します。 <code>libcurses(3LIB)</code> を参照してください。 |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Intro(3)

	libform	forms ライブラリ libform を構成する関数です。このライブラリは共有オブジェクト libform.so 、アーカイブ libforms.a として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 cc コマンド行で -lform を指定します。 libform(3LIB) を参照してください。
	libmenu	menus ライブラリ libmenu を構成する関数です。このライブラリは共有オブジェクト libmenu.so 、アーカイブ libmenu.a として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 cc コマンド行で -lmenu を指定します。 libmenu(3LIB) を参照してください。
	libpanel	panels ライブラリ libpanel を構成する関数です。このライブラリは共有オブジェクト libpanel.so として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 cc コマンド行で -lpanel を指定します。 libpanel(3LIB) を参照してください。
(3PLOT)		グラフィックスライブラリ libplot を構成する関数です。このライブラリは共有オブジェクト libplot.so 、アーカイブ libplot.a として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 cc コマンド行で -lplot を指定します。 libplot(3LIB) を参照してください。
(3XCURSES)		<code>/usr/xpg4/lib/libcurses.so.1</code> にある X/Open Curses ライブラリを構成する関数です。このライブラリは、端末画面での入力と出力、その変更を使用する国際化された関数とマクロのセットを提供します。このライブラリには、ウィンドウの作成、テキストの強調表示、画面への書き込み、ユーザーの入力の読み取り、カーソルの移動などを実行するための関数が含まれています。X/Open Curses は、画面更新動作を最適化するように設計されています。X/Open Curses ライブラリは、X/Open Extended Curses 仕様 (Issue 4) に全面的に準拠しています。
スレッドおよびリアルタイムライブラリ関数		このボリュームで説明する関数は、スレッドおよびリアルタイムライブラリを構成します。
(3AIO)		非同期入出力ライブラリ liaio を構成する関数です。このライブラリは共有オブジェクト libaio.so として実装されていますが、C コンパイルシステムによって自動的にリンクされませ

- ん。このライブラリにリンクするには、cc コマンド行で `-laio` を指定します。libaio(3LIB) を参照してください。
- (3DOOR) doors ライブラリ libdoor を構成する関数です。このライブラリは共有オブジェクト libdoor.so として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で `-ldoor` を指定します。
- (3RT) POSIX.4 リアルタイムライブラリ librt を構成する関数です。このライブラリは共有オブジェクト librt.so として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で `-lrt` を指定します。このライブラリは、以前は libposix4 という名前でした。下位互換のため、この libposix4 も維持されていますが、こちらの使用はできるだけ避けてください。librt(3LIB) を参照してください。
- (3SCHED) LWP スケジューリングライブラリ libsched を構成する関数です。このライブラリは共有オブジェクト libsched.so として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で `-lsched` を指定します。
- (3THR) スレッドライブラリ libpthread、libthread、libthread_db を構成する関数です。libpthread ライブラリと libthread ライブラリは、マルチスレッドアプリケーションの構築に使用します。libpthread は POSIX (standards(5) を参照) スレッドインタフェースを実装していますが、libthread が実装しているのは Solaris スレッドインタフェースです。libthread_db ライブラリは、マルチスレッドアプリケーションのデバッグの構築に役立ちます。
- POSIX スレッドと Solaris スレッドの実装には互換性があります。このため、双方のスレッドを同じアプリケーション内で使用することができます。ただし、POSIX 準拠の別の環境への移植性が保証されているのは POSIX スレッドの方だけです。
- POSIX スレッドと Solaris スレッドを同じアプリケーション内で使用しているとき、同じ名前でありながら意味解釈の異なる呼び出しがあると、Solaris スレッドではなく POSIX スレッドを指すものと解釈されます。たとえば、`fork()` の呼び出しは、POSIX スレッドライブラリにリンクしたプログラム内の `fork1()` を意味することになります。このプログラムが POSIX スレッドと同時に Solaris スレッド `-lthread` にもリンクしていたとしても、呼び出しが示しているのは POSIX スレッドの方になります。

libpthread、libthread、libthread_db の各ライブラリは、それぞれ共有オブジェクト libpthread.so、libthread.so、libthread_db.so として実装されていますが、アーカイブライブラリとしても実装されているのは libthread_db のみです。このアーカイブの名前は libthread_db.a です。これらのライブラリはいずれも、C コンパイラシステムによって自動的に実装されてはいません。このライブラリ群にリンクするには、cc コマンド行で、-lpthread、-lthread、-lthread_db のいずれかを指定します。libpthread(3LIB)、libthread(3LIB)、libthread_db(3LIB) を参照してください。

次の関数は、POSIX で使用するオプションであり、現在の Solaris リリースではサポートされていません。

```
int pthread_mutexattr_setprotocol(pthread_mutexattr_t *attr,
                                  int protocol);

int pthread_mutexattr_getprotocol(const pthread_mutexattr_t
                                   *attr,
                                   int *protocol);

int pthread_mutexattr_setprioceiling(pthread_mutexattr_t *attr,
                                       int prioceiling);

int pthread_mutexattr_getprioceiling
(const pthread_mutexattr_t *attr,
 int *prioceiling);
```

拡張ライブラリ関数

このボリュームで説明する関数は、特化されたさまざまなライブラリを構成します。構成されるライブラリは、次に示す限りではありません。

- (3BSM) 基本セキュリティライブラリ libbsm を構成する関数です。このライブラリは共有オブジェクト libbsm.so、アーカイブ libbsm.a として実装されていますが、C コンパイラシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で -lbsm を指定します。libbsm(3LIB) を参照してください。
- (3CFGADM) 構成管理ライブラリ libcfgadm を構成する関数です。このライブラリは共有オブジェクト libcfgadm.so として実装されていますが、C コンパイラシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で -lcfgadm を指定します。libcfgadm(3LIB) を参照してください。
- (3CPC) CPU パフォーマンスカウンタライブラリ libcpc およびプロセスコンテキストライブラリ libpctx を構成する関数です。これらのライブラリはそれぞれ共有オブジェクト libcpc.so および libpctx.so として実装されていますが、C コンパイラシステムによって自動的にリンクされません。このライブラリにリンク

	<p>するには、cc コマンド行で <code>-lcpc</code> を指定します。libcpc(3LIB) および libpctx(3LIB) を参照してください。</p>
(3DEVID)	<p>デバイス ID ライブラリ libdevid を構成する関数です。このライブラリは共有オブジェクト libdevid.so として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-ldevid</code> を指定します。libdevid(3LIB) を参照してください。</p>
(3DEVINFO)	<p>デバイス情報ライブラリ libbsm を構成する関数です。このライブラリは共有オブジェクト libdevinfo.so、アーカイブ libdevinfo.a として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-ldevinfo</code> を指定します。libdevinfo(3LIB) を参照してください。</p>
(3DMI)	<p>DMI ライブラリ libdmi、libdmici、libdmimi を構成する関数です。これらのライブラリはそれぞれ、共有オブジェクト libdmi.so、libdmici.so、libdmimi.so として実装されていますが、C コンパイルシステムによって自動的にリンクされません。これらにリンクするには、cc コマンド行で <code>-ldmi</code>、<code>-ldmici</code>、<code>-ldmimi</code> のいずれかを指定します。libdmi(3LIB)、libdmici(3LIB)、libdmimi(3LIB) を参照してください。</p>
(3ELF)	<p>ELF (Extensible Linking Format) アクセスライブラリ libelf を構成する関数です。このライブラリは elf ファイルの作成と分析用のインタフェースを提供します。elf ファイルには、実行ファイル、オブジェクト、共有オブジェクトがあります。libelf は共有オブジェクト libelf.so、アーカイブ libelf.a として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-lelf</code> を指定します。libelf(3LIB) を参照してください。</p>
(3EXACCT)	<p>拡張アカウントアクセスライブラリ libexacct およびプロジェクトデータベースアクセスライブラリ libproject を構成する関数です。これらのライブラリはそれぞれ、共有オブジェクト libexacct.so および libproject.so として実装されていますが、C コンパイルシステムによって自動的にリンクされません。これらのライブラリにリンクするには、cc コマンド行で <code>-lexacct</code> または <code>-lproject</code> を指定します。libexacct(3LIB) および libproject(3LIB) を参照してください。</p>
(3GEN)	<p>文字列のパターンマッチングやパス名の操作を行うライブラリ libgen を構成する関数です。このライブラリは共有オブジェクト libgen.so、アーカイブ libgen.a として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-lgen</code> を指定します。libgen(3LIB) を参照してください。</p>

Intro(3)

- | | |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| (3KSTAT) | カーネル統計情報ライブラリを構成する関数です。このライブラリは共有オブジェクト <code>libkstat.so</code> 、アーカイブ <code>libkstat.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 <code>cc</code> コマンド行で <code>-lkstat</code> を指定します。 <code>libkstat(3LIB)</code> を参照してください。 |
| (3KVM) | カーネルの仮想メモリーライブラリへのアクセスを許可する関数です。このライブラリは共有オブジェクト <code>libkvm.so</code> 、アーカイブ <code>libkvm.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 <code>cc</code> コマンド行で <code>-lkvm</code> を指定します。 <code>libkvm(3LIB)</code> を参照してください。 |
| (3LAYOUT) | レイアウトサービスライブラリを構成する関数です。このライブラリは共有オブジェクト <code>liblayout.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 <code>cc</code> コマンド行で <code>-llayout</code> を指定します。 <code>liblayout(3LIB)</code> を参照してください。 |
| (3LGRP) | ローカリティグループライブラリを構成する関数です。このライブラリは共有オブジェクト <code>liblgrp.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 <code>cc</code> コマンド行で <code>-llgrp</code> を指定します。 <code>liblgrp(3LIB)</code> を参照してください。 |
| (3M) | 数学ライブラリ <code>libm</code> を構成する関数です。このライブラリは共有オブジェクト <code>libm.so</code> 、アーカイブ <code>libm.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 <code>cc</code> コマンド行で <code>-lm</code> を指定します。 |
| (3MAIL) | ユーザーのメールボックス管理ライブラリ <code>libmail</code> を構成する関数です。このライブラリは共有オブジェクト <code>libmail.so</code> 、アーカイブ <code>libmail.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 <code>cc</code> コマンド行で <code>-lmail</code> を指定します。 |
| (3MP) | 整数型数学ライブラリ <code>libmp</code> を構成する関数です。このライブラリは共有オブジェクト <code>libmp.so</code> 、アーカイブ <code>libmp.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、 <code>cc</code> コマンド行で <code>-lmp</code> を指定します。 <code>libmp(3LIB)</code> を参照してください。 |
| (3NVP AIR) | 名前と値の組み合わせのライブラリ <code>libnvpair</code> を構成する関数です。このライブラリは共有オブジェクト <code>libnvpair.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。 |

	<p>リンクされません。このライブラリにリンクするには、<code>cc</code> コマンド行で <code>-lnvpair</code> を指定します。<code>libnvpair(3LIB)</code> を参照してください。</p>
(3PAM)	<p>PAM (Pluggable Authentication Module) ライブラリ <code>libpam</code> を構成する関数です。このライブラリは共有オブジェクト <code>libpam.so</code>、アーカイブ <code>libpam.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、<code>cc</code> コマンド行で <code>-lpam</code> を指定します。<code>libpam(3LIB)</code> を参照してください。</p>
(3PICL)	<p>PICL ライブラリ <code>libpicl</code> を構成する関数です。このライブラリは共有オブジェクト <code>libpicl.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、<code>cc</code> コマンド行で <code>-lpicl</code> を指定します。<code>libpicl(3LIB)</code>、<code>libpicl(3LIB)</code> および <code>libpicl(3PICL)</code> を参照してください。</p>
(3PICLTREE)	<p>PICL プラグインライブラリ <code>libpicltree</code> を構成する関数です。このライブラリは共有オブジェクト <code>libpicltree.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、<code>cc</code> コマンド行で <code>-lpicltree</code> を指定します。<code>libpicltree(3LIB)</code>、<code>libpicltree(3LIB)</code> および <code>libpicltree(3PICLTREE)</code> を参照してください。</p>
(3RSM)	<p>リモート共有メモリー <code>librsm</code> を構成する関数です。このライブラリは共有オブジェクト <code>librsm.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、<code>cc</code> コマンド行で <code>-lrsm</code> を指定します。<code>librsm(3LIB)</code> を参照してください。</p>
(3SEC)	<p>ファイルアクセス制御ライブラリ <code>libsec</code> を構成する関数です。このライブラリは共有オブジェクト <code>libsec.so</code>、アーカイブ <code>libsec.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、<code>cc</code> コマンド行で <code>-lsec</code> を指定します。<code>libsec(3LIB)</code> を参照してください。</p>
(3SECDB)	<p>セキュリティ属性データベースライブラリ <code>libsecdb</code> を構成する関数です。このライブラリは共有オブジェクト <code>libsecdb.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、<code>cc</code> コマンド行で <code>-lsecdb</code> を指定します。<code>libsecdb(3LIB)</code> を参照してください。</p>
(3SNMP)	<p>SNMP ライブラリ <code>libdssagent</code> と <code>libdssasmp</code> を構成する関数です。この2つのライブラリは、それぞれ共有オブジェクト <code>libssagent.so</code>、<code>libssasmp.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされませ</p>

Intro(3)

	<p>ん。これらにリンクするには、cc コマンド行で <code>-lssagent</code> または <code>-lssasmp</code> を指定します。libssagent(3LIB) を参照してください。</p>
(3SYSEVENT)	<p>システムイベントライブラリ <code>libsysevent</code> を構成する関数です。このライブラリは共有オブジェクト <code>libsysevent.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-lsysevent</code> を指定します。libsysevent(3LIB) を参照してください。</p>
(3TNF)	<p>TNF ライブラリ <code>libtnf</code>、<code>libtnfctl</code>、<code>libtnfprobe</code> を構成する関数です。この3つのライブラリは、それぞれ共有オブジェクト <code>libtnf.so</code>、<code>libtnfctl.so</code>、<code>libtnfprobe.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。これらにリンクするには、cc コマンド行で <code>-ltnf</code>、<code>-ltnfctl</code>、<code>-ltnfprobe</code> のいずれかを指定します。libtnfctl(3TNF) および libtnfctl(3LIB) を参照してください。</p>
(3VOLMGT)	<p>ボリューム管理ライブラリ <code>libvolmgt</code> を構成する関数です。このライブラリは共有オブジェクト <code>libvolmgt.so</code>、アーカイブ <code>libvolmgt.a</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-lvolmgt</code> を指定します。libvolmgt(3LIB) を参照してください。</p>
(3WSREG)	<p>製品インストールレジストリのライブラリ <code>libwsreg</code> を構成する関数です。このライブラリは、共有オブジェクト <code>libwsreg.so</code> として実装されていますが、C コンパイルシステムによって自動的にリンクされません。このライブラリにリンクするには、cc コマンド行で <code>-lwsreg</code> オプションを指定してください。libwsreg(3LIB) を参照してください。</p>
定義	<p>マシンのバイトに適合するビットパターンを、1文字と解釈することができます。ただし、言語によっては、1文字に1バイト以上必要で、複数バイトで表現されるものもあります。</p> <p>NULL 文字とは、値が0の文字のことです。C 言語では、便宜上この文字を <code>\0</code> と表します。文字の並び(シーケンス)のことを文字配列と言います。最後の文字が NULL 文字になっている文字の並びのことを NULL 終了文字配列(文字列)と言います。また、NULL 文字だけを含む文字配列のことを NULL 文字列と言います。0 をポインタにキャストすることによって得られる値は NULL ポインタと呼ばれています。C 言語では、この値は有効なポインタと一致しません。このため、ポインタを返す関数のほとんどは、NULL を返すことによってエラーを示します。NULL マクロは <code><stdio.h></code> で定義されています。size_t の型は適切なヘッダーの中に定義されています。</p>
MT レベルのライブラリファイル	<p>MT レベルのライブラリの詳細については <code>attributes(5)</code> を参照してください。</p> <p><code>INCDIR</code> 通常 <code>/usr/include</code></p>

	<code>LIBDIR</code>	通常 <code>/usr/lib</code> (32 ビット) または <code>/usr/lib/sparcv9</code> (64 ビット)
	<code>LIBDIR/libc.so</code>	
	<code>LIBDIR/libc.a</code>	
	<code>LIBDIR/libgen.a</code>	
	<code>LIBDIR/libm.a</code>	
	<code>LIBDIR/libsfm.sa</code>	
	<code>/usr/lib/libc.so.1</code>	
関連項目	<code>ar(1)</code> 、 <code>cc(1B)</code> 、 <code>ld(1)</code> 、 <code>fork(2)</code> 、 <code>intro(3)</code> 、 <code>stdio(3C)</code> 、 <code>attributes(5)</code> 、 <code>standards(5)</code>	
	『リンカーとライブラリ』	
	『プロファイリングツール』	
	『ANSI C Programmer's Guide』	
診断	浮動小数点の値を返す関数の場合、エラー処理はコンパイルのモードによって異なります。 <code>cc</code> に <code>-xt</code> オプションが指定されている場合 (デフォルト) に、指定された引数 が使用する関数が未定義であったり値を表現することができなかつたりすると、これ らの関数は <code>0</code> 、 <code>±HUGE</code> 、 <code>NaN</code> という一定の値を返します。 <code>-xa</code> や <code>-xc</code> のコンパイル モードでは、 <code>±HUGE</code> の代わりに <code>±HUGE_VAL</code> が返されます。 <code>HUGE_VAL</code> と <code>HUGE</code> は <code>math.h</code> の中で定義されています。 <code>HUGE_VAL</code> は値が無限であることを表し、 <code>HUGE</code> は単精度小数点の最大値を表します。	
マルチスレッドア プリケーションに 関する注意事項	マルチスレッドアプリケーションをコンパイルする際は、コマンド行で <code>_POSIX_C_SOURCE</code> フラグ、 <code>_POSIX_PTHREAD_SEMANTICS</code> フラグ、 <code>_REENTRANT</code> フラグのいずれかのフラグを定義する必要があります。これにより、マルチスレッド のアプリケーションだけに適用される関数の定義が特別に有効になります。 <code>POSIX.1003.1c</code> に適合するアプリケーションには、 <code>_POSIX_C_SOURCE</code> フラグに <code>199506L</code> 以上の値を定義してください。	
	<pre>cc [flags] file. . . -D_POSIX_C_SOURCE=199506L -lpthread</pre>	
	Solaris の <code>fork()</code> と <code>fork1()</code> 関数を使用した POSIX スレッドは、次のようにコン パイルしてください。	
	<pre>cc [flags] file. . . -D_POSIX_PTHREAD_SEMANTICS -lthread</pre>	
	Solaris スレッドは、次のようにコンパイルしてください。	
	<pre>cc [flags] file... -D_REENTRANT -lthread</pre>	

Intro(3)

リアルタイムアプリケーション

注意事項

シングルスレッドのアプリケーションを作成する際、上記のフラグを定義してはいけません。定義しないことにより、マルチスレッドをサポートしていない旧リリースの Solaris 上で実行可能なバイナリが生成されます。

アプリケーションを安全に実行するため、メインスレッド以外からは `Unsafe` インタフェースを呼び出さないようにしてください。

MT-Safe インタフェースは、各関数やライブラリのマニュアルページの「属性」の項目に記述されています (`attributes(5)` を参照)。インタフェースが MT-Safe であることがマニュアルページ中に明示されていないときは、そのインタフェースを `Unsafe` とみなしてください。

早期のバインディングを可能にするためには、環境変数 `LD_BIND_NOW` の値を非 NULL 値に設定してください。詳細については『リンカーとライブラリ』の再配置に関する説明を参照してください。

ここで説明する関数、外部変数、マクロは、ユーザープログラムの中で再定義しないでください。他の名前を再定義してもこれらのライブラリの動作に影響を与えませんが、同じ名前を再定義すると、インクルードされているヘッダーの宣言と衝突してしまいます。

`INCDIR` 中にあるヘッダーは、このマニュアルページで説明するほとんどの関数のプロトタイプを提供します。関数のプロトタイプとは、引数の型を含む関数宣言のことです。関数のプロトタイプを使用すると、ユーザーのプログラムの中で使用されている関数の使用法が間違っていないか、コンパイラでチェックすることができます。コンパイラの代わりに `lint` プログラムチェッカーを使用することもできます。`lint` は、ヘッダーが `#include` 文でインクルードされていないことを検出すると、この矛盾を報告します。セクション 2、3C、3S の定義は自動的にチェックされます。その他の定義をチェックするには、`lint` に `-l` オプションを指定します。たとえば、`libm` の定義をチェックするには `-lm` を指定します。できるだけ `lint` を使用するようしてください。詳細は、『プロファイリングツール』の `lint` に関する章を参照してください。

「STREAMS」と「ストリーム」の違いには十分気を付けてください。`STREAMS` とは、ネットワークサービスとデータ通信ドライバの開発をサポートするカーネル機構の一種であり、ユーティリティルーチン、カーネル機能、データ構造体で構成されています。一方、ストリームとは、`STREAMS` に関連するバッファリングで使用されるファイルのことです。ストリームは `<stdio.h>` の中で `FILE` 型へのポインタとして定義されています。

各要素の詳細な定義では、実装に固有のシンボリック名を参照しなければならない場合があります。ただし、必ずしもアプリケーションプログラムで使用できるようにする必要はありません。こうしたシンボリック名の多くは、境界の条件やシステムの制限を記述するものです。

このセクションでは、分かりやすくするために、このような実装固有の値をシンボリック名にしています。このようなシンボリック名は必ず中括弧 (`{}`) で囲まれ、ヘッダーを使ってアプリケーションからアクセスできる実装固有の定数のシンボ

リック名と区別されます。中括弧で囲まれているシンボリック名は、特定のシステムの文書中で定義されていることもあります。ヘッダーを使用することによってアプリケーションプログラムから必ずしもアクセスできるわけではありません。

一般には、移植可能なアプリケーションプログラムコードの中で、このようなシンボリック名が参照されてはなりません。たとえば、あるアプリケーションプログラムで、あるルーチンに提供された引数リストの長さが {ARG_MAX} より大きいかどうかをテストしてはなりません。

X/Open XFN 標準に基づいたフェデレーテッドネーミングサービスは、Solaris オペレーティングシステムの将来のリリースではサポートされない可能性があります。

コマンド一覧

名前	説明
Intro(3)	関数とライブラリの紹介
addnwstr(3curses)	curl_addwstr(3CURSES) を参照
addwch(3curses)	curl_addwch(3CURSES) を参照
addwchnstr(3curses)	curl_addwchstr(3CURSES) を参照
addwchstr(3curses)	curl_addwchstr(3CURSES) を参照
addwstr(3curses)	curl_addwstr(3CURSES) を参照
adjcurspos(3curses)	curl_alecompat(3CURSES) を参照
advance(3gen)	regexpr(3GEN) を参照
ascftime(3c)	strftime(3C) を参照
bindtextdomain(3c)	gettext(3C) を参照
bind_textdomain_codeset(3c)	gettext(3C) を参照
cftime(3c)	strftime(3C) を参照
compile(3gen)	regexpr(3GEN) を参照
config_admin(3CFGADM)	構成の管理
config_ap_id_cmp(3cfgadm)	config_admin(3CFGADM) を参照
config_change_state(3cfgadm)	config_admin(3CFGADM) を参照
config_list(3cfgadm)	config_admin(3CFGADM) を参照
config_list_ext(3cfgadm)	config_admin(3CFGADM) を参照
config_private_func(3cfgadm)	config_admin(3CFGADM) を参照
config_stat(3cfgadm)	config_admin(3CFGADM) を参照
config_strerror(3cfgadm)	config_admin(3CFGADM) を参照
config_test(3cfgadm)	config_admin(3CFGADM) を参照
config_unload_libs(3cfgadm)	config_admin(3CFGADM) を参照

Intro(3)

<code>cset(3C)</code>	EUC コードセットに関する情報の取得
<code>csetcol(3c)</code>	<code>cset(3C)</code> を参照
<code>csetlen(3c)</code>	<code>cset(3C)</code> を参照
<code>csetno(3c)</code>	<code>cset(3C)</code> を参照
<code>ctype(3C)</code>	文字処理
<code>curses(3CURSES)</code>	CRT 画面操作および最適化用パッケージ
<code>curl_addwch(3CURSES)</code>	<code>wchar_t</code> 文字を属性とともに <code>curses</code> ウィンドウに追加してカーソルを進める
<code>curl_addwchstr(3CURSES)</code>	<code>wchar_t</code> 文字列を属性とともに <code>curses</code> ウィンドウに追加
<code>curl_addwstr(3CURSES)</code>	<code>wchar_t</code> 文字列を <code>curses</code> ウィンドウに追加してカーソルを進める
<code>curl_alecompat(3CURSES)</code>	カーソルを文字単位に移動するために ALE <code>curses</code> ライブラリに追加された関数
<code>curl_getwch(3CURSES)</code>	<code>curses</code> 端末キーボードから <code>wchar_t</code> 文字を得る (または戻す)
<code>curl_getwstr(3CURSES)</code>	<code>curses</code> 端末キーボードから <code>wchar_t</code> 文字列を得る
<code>curl_inswch(3CURSES)</code>	<code>curses</code> ウィンドウ内のカーソル位置の文字の直前に <code>wchar_t</code> 文字を挿入
<code>curl_inswstr(3CURSES)</code>	<code>curses</code> ウィンドウ内のカーソル位置の文字の直前に <code>wchar_t</code> 文字列を挿入
<code>curl_inwch(3CURSES)</code>	<code>curses</code> ウィンドウから <code>wchar_t</code> 文字とその属性を得る
<code>curl_inwchstr(3CURSES)</code>	<code>curses</code> ウィンドウから <code>wchar_t</code> 文字列とその属性を得る
<code>curl_inwstr(3CURSES)</code>	<code>curses</code> ウィンドウから <code>wchar_t</code> 文字列を得る
<code>curl_pad(3CURSES)</code>	<code>curses</code> パッドの作成と表示
<code>dcgettext(3c)</code>	<code>gettext(3C)</code> を参照
<code>dcngettext(3c)</code>	<code>gettext(3C)</code> を参照
<code>dgettext(3c)</code>	<code>gettext(3C)</code> を参照
<code>dngettext(3c)</code>	<code>gettext(3C)</code> を参照
<code>echowchar(3curses)</code>	<code>curl_addwch(3CURSES)</code> を参照
<code>euccol(3c)</code>	<code>euclen(3C)</code> を参照

euclen(3C)	EUC 文字のバイト長および表示幅の取得
eucscol(3c)	euclen(3C) を参照
fgetwc(3C)	ストリームからワイド文字コードを読み込む
fgetws(3c)	getws(3C) を参照
fprintf(3c)	printf(3C) を参照
fputwc(3C)	ワイド文字コードをストリームへ書き出す
fputws(3C)	ワイド文字列をストリームへ書き出す
fscanf(3c)	scanf(3C) を参照
getdate(3C)	ユーザー形式の日付および時刻の変換
getnwstr(3curses)	curl_getwstr(3CURSES) を参照
gettext(3C)	メッセージ処理関数
getwc(3C)	ストリームからワイド文字を読み込む
getwch(3curses)	curl_getwch(3CURSES) を参照
getwchar(3C)	stdin ストリームからワイド文字を読み込み
getwidth(3C)	コードセット情報の取得
getws(3C)	ストリームからワイド文字列を取得する
getwstr(3curses)	curl_getwstr(3CURSES) を参照
iconv(3C)	コード変換関数
innwstr(3curses)	curl_inwstr(3CURSES) を参照
insnwstr(3curses)	curl_inswstr(3CURSES) を参照
inswch(3curses)	curl_inswch(3CURSES) を参照
inswstr(3curses)	curl_inswstr(3CURSES) を参照
inwch(3curses)	curl_inwch(3CURSES) を参照
inwchnstr(3curses)	curl_inwchstr(3CURSES) を参照
inwchstr(3curses)	curl_inwchstr(3CURSES) を参照
inwstr(3curses)	curl_inwstr(3CURSES) を参照
isalnum(3c)	ctype(3C) を参照
isalpha(3c)	ctype(3C) を参照
isascii(3c)	ctype(3C) を参照
isctrl(3c)	ctype(3C) を参照

Intro(3)

isdigit(3c)	ctype(3C) を参照
isencrypt(3GEN)	文字のバッファが暗号化されているかの判別
isenglish(3c)	iswalpha(3C) を参照
isgraph(3c)	ctype(3C) を参照
isideogram(3c)	iswalpha(3C) を参照
islower(3c)	ctype(3C) を参照
isnumber(3c)	iswalpha(3C) を参照
isphonogram(3c)	iswalpha(3C) を参照
isprint(3c)	ctype(3C) を参照
ispunct(3c)	ctype(3C) を参照
isspace(3c)	ctype(3C) を参照
isspecial(3c)	iswalpha(3C) を参照
isupper(3c)	ctype(3C) を参照
iswalnum(3c)	iswalpha(3C) を参照
iswalpha(3C)	ワイド文字の文字分類用関数
iswascii(3c)	iswalpha(3C) を参照
iswcntrl(3c)	iswalpha(3C) を参照
iswctype(3C)	指定された文字のクラスの検査
iswdigit(3c)	iswalpha(3C) を参照
iswgraph(3c)	iswalpha(3C) を参照
iswlower(3c)	iswalpha(3C) を参照
iswprint(3c)	iswalpha(3C) を参照
iswpunct(3c)	iswalpha(3C) を参照
iswspace(3c)	iswalpha(3C) を参照
iswupper(3c)	iswalpha(3C) を参照
iswxdigit(3c)	iswalpha(3C) を参照
isxdigit(3c)	ctype(3C) を参照
libcfgadm(3LIB)	構成管理のライブラリ
mblen(3C)	文字のバイト数を取得する
mbstowcs(3C)	文字列をワイド文字列に変換

<code>mbtowc(3C)</code>	複数バイト文字をワイド文字コードに変換
<code>movenextch(3curses)</code>	<code> curs_alecompat(3CURSES)</code> を参照
<code>moveprevch(3curses)</code>	<code> curs_alecompat(3CURSES)</code> を参照
<code>mvaddnwstr(3curses)</code>	<code> curs_addwstr(3CURSES)</code> を参照
<code>mvaddwch(3curses)</code>	<code> curs_addwch(3CURSES)</code> を参照
<code>mvaddwchnstr(3curses)</code>	<code> curs_addwchstr(3CURSES)</code> を参照
<code>mvaddwchstr(3curses)</code>	<code> curs_addwchstr(3CURSES)</code> を参照
<code>mvaddwstr(3curses)</code>	<code> curs_addwstr(3CURSES)</code> を参照
<code>mvgetnwstr(3curses)</code>	<code> curs_getwstr(3CURSES)</code> を参照
<code>mvgetwch(3curses)</code>	<code> curs_getwch(3CURSES)</code> を参照
<code>mvgetwstr(3curses)</code>	<code> curs_getwstr(3CURSES)</code> を参照
<code>mvinnwstr(3curses)</code>	<code> curs_inwstr(3CURSES)</code> を参照
<code>mvinsnwstr(3curses)</code>	<code> curs_inswstr(3CURSES)</code> を参照
<code>mvinswch(3curses)</code>	<code> curs_inswch(3CURSES)</code> を参照
<code>mvinswstr(3curses)</code>	<code> curs_inswstr(3CURSES)</code> を参照
<code>mvinwch(3curses)</code>	<code> curs_inwch(3CURSES)</code> を参照
<code>mvinwchnstr(3curses)</code>	<code> curs_inwchstr(3CURSES)</code> を参照
<code>mvinwchstr(3curses)</code>	<code> curs_inwchstr(3CURSES)</code> を参照
<code>mvinwstr(3curses)</code>	<code> curs_inwstr(3CURSES)</code> を参照
<code>mvwaddnwstr(3curses)</code>	<code> curs_addwstr(3CURSES)</code> を参照
<code>mvwaddwch(3curses)</code>	<code> curs_addwch(3CURSES)</code> を参照
<code>mvwaddwchnstr(3curses)</code>	<code> curs_addwchstr(3CURSES)</code> を参照
<code>mvwaddwchstr(3curses)</code>	<code> curs_addwchstr(3CURSES)</code> を参照
<code>mvwaddwstr(3curses)</code>	<code> curs_addwstr(3CURSES)</code> を参照
<code>mvwgetnwstr(3curses)</code>	<code> curs_getwstr(3CURSES)</code> を参照
<code>mvwgetwch(3curses)</code>	<code> curs_getwch(3CURSES)</code> を参照
<code>mvwgetwstr(3curses)</code>	<code> curs_getwstr(3CURSES)</code> を参照
<code>mvwinnwstr(3curses)</code>	<code> curs_inwstr(3CURSES)</code> を参照
<code>mvwinsnwstr(3curses)</code>	<code> curs_inswstr(3CURSES)</code> を参照
<code>mvwinswch(3curses)</code>	<code> curs_inswch(3CURSES)</code> を参照
<code>mvwinswstr(3curses)</code>	<code> curs_inswstr(3CURSES)</code> を参照

Intro(3)

<code>mvwinwch(3curses)</code>	<code> curs_inwch(3CURSES)</code> を参照
<code>mvwinwchnstr(3curses)</code>	<code> curs_inwchstr(3CURSES)</code> を参照
<code>mvwinwchstr(3curses)</code>	<code> curs_inwchstr(3CURSES)</code> を参照
<code>mvwinwstr(3curses)</code>	<code> curs_inwstr(3CURSES)</code> を参照
<code>newpad(3curses)</code>	<code> curs_pad(3CURSES)</code> を参照
<code>ngettext(3c)</code>	<code> gettext(3C)</code> を参照
<code>pechochar(3curses)</code>	<code> curs_pad(3CURSES)</code> を参照
<code>pechowchar(3curses)</code>	<code> curs_pad(3CURSES)</code> を参照
<code>pnoutrefresh(3curses)</code>	<code> curs_pad(3CURSES)</code> を参照
<code>prefresh(3curses)</code>	<code> curs_pad(3CURSES)</code> を参照
<code>printf(3C)</code>	書式付き出力
<code>putwc(3c)</code>	<code> fputwc(3C)</code> を参照
<code>putwchar(3c)</code>	<code> fputwc(3C)</code> を参照
<code>putws(3C)</code>	ワイド文字列の EUC 文字への変換
<code>regexpr(3GEN)</code>	正規表現のコンパイルおよび一致ルーチン
<code>scanf(3C)</code>	書式付き入力の変換
<code>setlocale(3C)</code>	プログラムのロケールの変更および照会
<code>snprintf(3c)</code>	<code> printf(3C)</code> を参照
<code>sprintf(3c)</code>	<code> printf(3C)</code> を参照
<code>sscanf(3c)</code>	<code> scanf(3C)</code> を参照
<code>step(3gen)</code>	<code> regexpr(3GEN)</code> を参照
<code>strftime(3C)</code>	日付と時刻を文字列に変換
<code>subpad(3curses)</code>	<code> curs_pad(3CURSES)</code> を参照
<code>textdomain(3c)</code>	<code> gettext(3C)</code> を参照
<code>toascii(3C)</code>	整数を7ビット ASCII 文字へ変換
<code>_tolower(3C)</code>	大文字を小文字に変換
<code>tolower(3C)</code>	大文字を小文字に変換
<code>_toupper(3C)</code>	小文字を大文字に変換
<code>toupper(3C)</code>	小文字を大文字に変換
<code>towlower(3C)</code>	ワイド文字表現の大文字を小文字に変換
<code>towupper(3C)</code>	ワイド文字表現の小文字を大文字に変換

ungetwc(3C)	ワイド文字コードを入力ストリームに プッシュバック
ungetwch(3curses)	curs_getwch(3CURSES) を参照
vfscanf(3c)	scanf(3C) を参照
vscanf(3c)	scanf(3C) を参照
vsscanf(3c)	scanf(3C) を参照
waddnwstr(3curses)	curs_addwstr(3CURSES) を参照
waddwch(3curses)	curs_addwch(3CURSES) を参照
waddwchnstr(3curses)	curs_addwchstr(3CURSES) を参照
waddwchstr(3curses)	curs_addwchstr(3CURSES) を参照
waddwstr(3curses)	curs_addwstr(3CURSES) を参照
wadjcurpos(3curses)	curs_alecompat(3CURSES) を参照
watof(3c)	wcstod(3C) を参照
watoi(3c)	wcstol(3C) を参照
watol(3c)	wcstol(3C) を参照
watoll(3c)	wcstol(3C) を参照
wscat(3c)	wcstring(3C) を参照
wschr(3c)	wcstring(3C) を参照
wscmp(3c)	wcstring(3C) を参照
wscoll(3C)	照合情報を使ってワイド文字列を比較
wscopy(3c)	wcstring(3C) を参照
wscspn(3c)	wcstring(3C) を参照
wcsetno(3c)	cset(3C) を参照
wcsftime(3C)	日付および時刻をワイド文字列に変換
wcslen(3c)	wcstring(3C) を参照
wcsncat(3c)	wcstring(3C) を参照
wcsncmp(3c)	wcstring(3C) を参照
wcsncpy(3c)	wcstring(3C) を参照
wcspbrk(3c)	wcstring(3C) を参照
wcsrchr(3c)	wcstring(3C) を参照
wcsspn(3c)	wcstring(3C) を参照

Intro(3)

<code>wcstod(3C)</code>	ワイド文字列を倍精度浮動小数点数に変換
<code>wcstok(3c)</code>	<code>wcstring(3C)</code> を参照
<code>wcstol(3C)</code>	ワイド文字列を整数に変換
<code>wcstombs(3C)</code>	ワイド文字列を文字列に変換
<code>wcstoul(3C)</code>	ワイド文字列を符号なしロング整数に変換
<code>wcstring(3C)</code>	ワイド文字列の操作
<code>wcswcs(3c)</code>	<code>wcstring(3C)</code> を参照
<code>wcswidth(3C)</code>	ワイド文字列のカラム数
<code>wcsxfrm(3C)</code>	ワイド文字列の変換
<code>wctomb(3C)</code>	ワイド文字コードを複数バイト文字に変換
<code>wctype(3C)</code>	文字クラスの定義
<code>wcwidth(3C)</code>	ワイド文字コードのカラム数
<code>wechowchar(3curses)</code>	<code>curs_addwch(3CURSES)</code> を参照
<code>wgetnwstr(3curses)</code>	<code>curs_getwstr(3CURSES)</code> を参照
<code>wgetwch(3curses)</code>	<code>curs_getwch(3CURSES)</code> を参照
<code>wgetwstr(3curses)</code>	<code>curs_getwstr(3CURSES)</code> を参照
<code>windex(3c)</code>	<code>wcstring(3C)</code> を参照
<code>winnwstr(3curses)</code>	<code>curs_inwstr(3CURSES)</code> を参照
<code>winsnwstr(3curses)</code>	<code>curs_inswstr(3CURSES)</code> を参照
<code>winswch(3curses)</code>	<code>curs_inswch(3CURSES)</code> を参照
<code>winswstr(3curses)</code>	<code>curs_inswstr(3CURSES)</code> を参照
<code>winwch(3curses)</code>	<code>curs_inwch(3CURSES)</code> を参照
<code>winwchnstr(3curses)</code>	<code>curs_inwchstr(3CURSES)</code> を参照
<code>winwchstr(3curses)</code>	<code>curs_inwchstr(3CURSES)</code> を参照
<code>winwstr(3curses)</code>	<code>curs_inwstr(3CURSES)</code> を参照
<code>wmovenextch(3curses)</code>	<code>curs_alecompat(3CURSES)</code> を参照
<code>wmoveprevch(3curses)</code>	<code>curs_alecompat(3CURSES)</code> を参照
<code>wrindex(3c)</code>	<code>wcstring(3C)</code> を参照
<code>wscasecmp(3c)</code>	<code>wstring(3C)</code> を参照
<code>wscat(3c)</code>	<code>wcstring(3C)</code> を参照
<code>wschr(3c)</code>	<code>wcstring(3C)</code> を参照

wscmp(3c)	wcstring(3C) を参照
wscol(3c)	wstring(3C) を参照
wscoll(3c)	wscoll(3C) を参照
wscopy(3c)	wcstring(3C) を参照
wscspn(3c)	wcstring(3C) を参照
wsdup(3c)	wstring(3C) を参照
wslen(3c)	wcstring(3C) を参照
wsncasecmp(3c)	wstring(3C) を参照
wsncat(3c)	wcstring(3C) を参照
wsncmp(3c)	wcstring(3C) を参照
wsncpy(3c)	wcstring(3C) を参照
wspbrk(3c)	wcstring(3C) を参照
wsprintf(3C)	書式付き出力変換
wsrchr(3c)	wcstring(3C) を参照
wsscanf(3C)	書式付き入力変換
wsspncpy(3c)	wcstring(3C) を参照
wstod(3c)	wcstod(3C) を参照
wstok(3c)	wcstring(3C) を参照
wstol(3c)	wcstol(3C) を参照
wstring(3C)	ワイド文字の文字列操作
wsxfrm(3c)	wcsxfrm(3C) を参照

libcfgadm(3LIB)

名前	libcfgadm – 構成管理のライブラリ						
形式	cc [<i>flag</i> . . .] <i>file</i> . . . -lcfgadm -ldevinfo -ldl [<i>library</i> . . .] #include <config_admin.h>						
機能説明	このライブラリに含まれるインタフェースは、構成管理のためのサービスを提供します。						
インタフェース	共有オブジェクト libcfgadm.so.1 は、以下に示す共通インターフェースを提供します。 共有オブジェクトのインタフェースについては、intro(3) を参照してください。						
	<pre> config_ap_id_cmp config_change_state config_help config_list config_list_ext config_private_func config_stat config_strerror config_test config_unload_libs </pre>						
ファイル	/usr/lib/libcfgadm.so.1 共有オブジェクトファイル /usr/lib/64/libcfgadm.so.1 64 ビット共有オブジェクトファイル						
属性	以下の属性については、attributes(5) を参照してください。						
	<table border="1"> <thead> <tr> <th>属性の種類</th> <th>属性の値</th> </tr> </thead> <tbody> <tr> <td>使用条件</td> <td>SUNWcsl (32-bit) SUNWcslx (64-bit)</td> </tr> <tr> <td>MT レベル</td> <td>Mt-Safe</td> </tr> </tbody> </table>	属性の種類	属性の値	使用条件	SUNWcsl (32-bit) SUNWcslx (64-bit)	MT レベル	Mt-Safe
属性の種類	属性の値						
使用条件	SUNWcsl (32-bit) SUNWcslx (64-bit)						
MT レベル	Mt-Safe						
関連項目	pvs(1), cfgadm(1M), intro(3), config_admin(3CFGADM), attributes(5)						