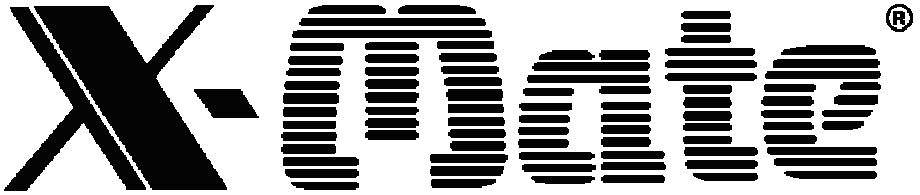


グラフィカル・ユーザー・インターフェース
構築ツール

開発ライブラリ

リファレンス・マニュアル

Version 3.0

はじめに

本マニュアルは、X Window System 上でXlibを使って作成するプログラムをより使いやすくする為に開発されたライブラリ"X-Mate開発ライブラリ"を使いこなす為のリファレンスマニュアルです。

本マニュアルは、プログラミングを行なう為に必要な各部品（オブジェクト）及び関数（サブルーチン）の構成等のみを、詳細に説明しています。その為、実際のプログラム例は記載されていませんので、『X-Mate開発ライブラリ プログラミングマニュアル』を参照して下さい。

本書をお読みになり、X-Mate開発ライブラリをご利用頂ければ幸いと存じます。

目次

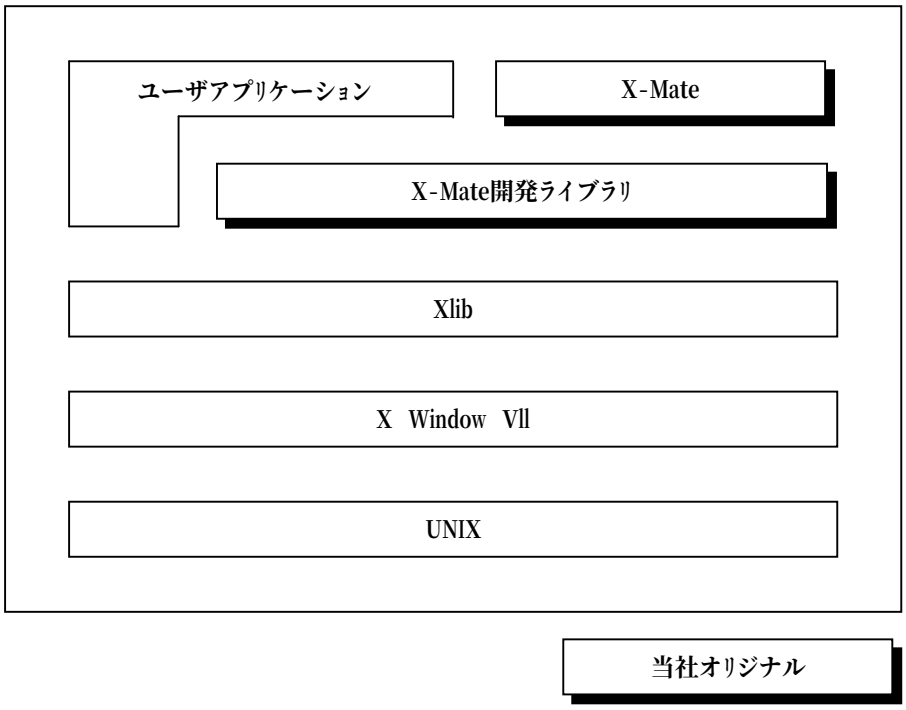
1 概要

X-Mate開発ライブラリは、X Window System 上で動作する画面を作成する為のGUI開発支援ライブラリです。

以下の特徴があります。

- 表示画面の保存
- 各種イベントのハンドリング
- ウィンドウの階層管理
- ウィンドウ間の通信機能
- 日本語入出力のサポート

ソフトウェア構成図



2 部品情報

X-Mate開発ライブラリの種類を下記のように三種類に分ける事が出来ます。

- 描画機能を持っている表示部品。
- ユーザとのインターフェイスを行なうイベント部品。
- ウィンドウの制御や部品の表示更新等を行なうサブルーチン。

本章では、表示部品とイベント部品について説明します。
(サブルーチンについては、3 関数情報を参照して下さい。)

表示部品

パネル上に文字を表示したり、円や線の表示を行ない、画面を形成します。

イベント部品

イベント部品には、ボタン、スクロールバー、キー入力等の部品があり、ユーザの操作(マウスクリック、キー入力)に対して、何らかのアクションを定義する事が出来ます。
この”何らかのアクションを定義する事が出来る”とは、部品にコールバックルーチンを登録し、自由にアプリケーションプログラムを作成出来るという事です。

2.1　部品共通情報
（オブジェクトヘッダ説明）

　各部品の定義データは、先頭に共通のヘッダ情報を持ち、その後に固有の部品情報を持ちます。
　部品を定義する時は、必ずヘッダ情報を設定して下さい。
　部品ヘッダ情報の説明を以下に示します。

構 造 体

```
typedef struct
{
    int         type;       /*    オブジェクトタイプ                    */
    QID         id;         /*    表示チェイン ID                       */
    int         flag;       /*    表示フラグ                            */
}   Ktobj;
```

メンバー説明

type　　オブジェクトタイプを設定する事で、X-Mate開発ライブラリがどの部品情報かを判断します。

オブジェクトのタイプを以下に示します。

モード	値	モード	値
テキスト	KOTEXT	テキスト入力	KOITEXT
リピートテキスト	KORTEXT	新テキスト	KONTEXT
ライン	KOLINE	データ入力	KOIDATA
罫線	KOKEI	データ 3	KODATA3
矩形	KORECT	スクロールバー	KOSBAR
円弧	KOARC	ポインタ	KOPOIT
ポリゴン	KOPOLY	ムーブ	KOMOVE
ビットマップ	KOIMAG	プルダウンメニュー	KOMENU
棒グラフ	KOBARG	新メニュー	KONMENU
バーカーソル	KOBARC	リスト	KOLIST
ボックスカーソル	KOBOXC	帳票	KOTABLE
ラスター	KORAST	新帳票	KOTABLE2
キャンバス	KOCANV	複合部品	KOCOMB
ボタン	KOBUTN		

id　　パネルに対して登録された時のチェインIDが設定されます。
ライブラリが管理する為 初期値はNULL を指定します。

flag　　部品登録制御を行なう際の機能を指定します。

　　1・・・通常登録。（初期値）
　　0・・・登録無効。

【部品サイズの仮想化】
　次の設定値を論理和にて追加する事により、部品登録後にウィンドウを変形させた際に、ウィンドウサイズに追従して部品サイズが変形する機能を持ちます。

　KLOGX　　:論理 X座標
　KLOGY　　:論理 Y座標
　KLOGW　　:論理 幅
　KLOGH　　:論理 高さ

＜設定例＞ button.hed.flag = 1 |KLOGX |KLOGY |KLOGW |KLOGH;

部品サイズの仮想化機能を指定した際の部品動作は次の通りです。

【テキスト部品】

文字の大きさは変化しません。
表示エリアの大きさが変形し、表示を可能な文字数が変化します。

【スクロールバーが付く部品】　（ラスター、帳票、新テキスト、キャンバス部品等）

表示エリアのみ変化します。

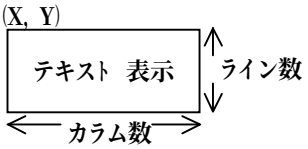
【その他の部品】

図形の描画サイズが変化します。

テキスト　KOTEXT

機　能

テキストデータの表示を行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  テキスト表示モード                  */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         c;          /*  表示カラム数                        */
    int         l;          /*  表示ライン数                        */
    char        *fc;        /*  テキスト表示色                      */
    char        *bc;        /*  テキスト背景色                      */
    int         font;       /*  使用フォントコード                  */
    char        *text;      /*  テキストストリングスアドレス        */
}   Kttext;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　テキストの表示形式を以下に示します。

モード	値	説　　明
重ね書き	KTOR	背景に重ね合わせて表示
上書き	KTSET	フォントの大きさを背景色で塗りつぶして表示
反転	KTREV	テキスト表示色と背景色を反転して表示
横書き	KTANG	90°フォントを回転して表示

x
y　　テキストの表示開始位置を示すウィンドウ上のポジションを示します。

c
l　　テキスト表示エリアの幅、行数を示します。

*fc
*bc　　表示を行なうテキストの文字色と背景色を示します。
表示モードが重ね書きの時、背景色の指定は無効になります。
(付録1⑤参照)

font　　テキストの表示に使用する文字フォントを示します。
(付録1④-1参照)

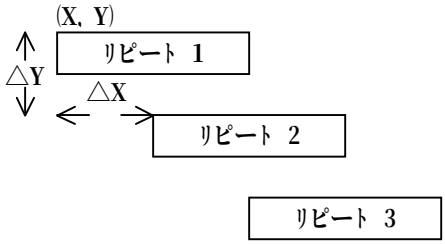
*text　　表示を行なうテキストのストリングスアドレスです。
ストリングス中に改行コードを挿入する事により、文字列を改行して
表示する事が出来ます。
また、表示エリアをはみだした部分のテキストは表示されません。

MEMO

リピートテキスト　　KORTEXT

機　能

テキストデータの等間隔表示を行ないます。



△X･･･X増加分値
△Y･･･Y増加分値

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         x;          /*  表示開始位置 X                      */
    int         y;          /*  表示開始位置 Y                      */
    int         dx;         /*  X増加値                             */
    int         dy;         /*  Y増加値                             */
    int         l;          /*  リピート回数                        */
    char        *bc;        /*  テキスト背景色                      */
    int         font;       /*  使用フォントコード                  */
    Ktrtdt      *tdat;      /*  テキストデータアドレス              */
}   Ktrtxt;


typedef struct
{
    int         mode;       /*  テキスト表示モード                  */
    char        *fc;        /*  テキスト表示色                      */
    char        *text;      /*  テキストストリングスアドレス        */
}   Ktrtdt;
```

メンバー説明

hed	オブジェクトのヘッダ情報を示します。 (2.1部品共通情報参照)
x y	テキストのリピート表示の位置を示します。
dx dy	次テキストの表示位置X, Yの増分を示します。
l	表示繰り返し回数を示します。
*bc	テキストの背景色を示します。 (付録1⑤参照)

font　　テキストの表示に使用する文字フォントを示します。
（付録1④-1参照）

*tdat　　テキストデータのアドレスを示します。
テキストデータはリピート回数分必要です。

mode　　テキストの表示形式を以下に示します。

モード	値	説　　明
重ね書き	KTOR	背景に重ね合わせて表示
上書き	KTSET	フォントの大きさを背景色で塗りつぶして表示
反転	KTREV	テキスト表示色と背景色を反転して表示
横書き	KTANG	90°フォントを回転して表示

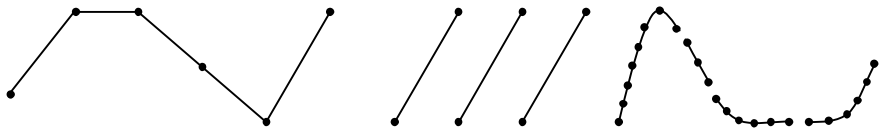
*fc　　テキストの文字色を示します。
（付録1⑤参照）

*text　　表示を行なうテキストのストリングスアドレスです。
また、表示エリアをはみだした部分のテキストは表示されません。

ライン　　KOLINE

機　能

プレート上の点を複数指定し、線を引きます。



継続モード　　単体モード　　グラフモード

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  ラインモード                        */
    int         func;       /*  描画ファンクション                  */
    int         pmode;      /*  座標モード                          */
    int         ptn;        /*  太さ/線種                           */
    char        *col;       /*  線色                                */
    int         pc;         /*  座標数                              */
    Ktptdt      *pt;        /*  座標データアドレス                  */
}   Ktline;

typedef struct              /*  座標データ構造体                    */
{
    short       x;          /*  座標  X                             */
    short       y;          /*  座標  Y                             */
}   Ktptdt;

typedef struct              /*  座標データ構造体                    */
{
    short       x;          /*  座標  X                             */
    short       y;          /*  座標  Y                             */
    short       flg;        /*  フラグ                              */
}   Ktptgr;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　ラインを描画するモードを以下に示します。

モード	値	説　　明
継続	KLMODE	全ての座標を結びラインを描画
単体	KLSING	2つの座標をペアにして線分を描画
グラフ	KLGRPH	フラグにより出力制御を行なう

func　　ラインを描画する際のグラフィック関数を指定します。
通常はGXcopyを指定します。
(付録1⑪参照)

pmode　　座標データの形式を以下に示します。(ラインモード継続時のみ
相対座標が使用出来ます。)

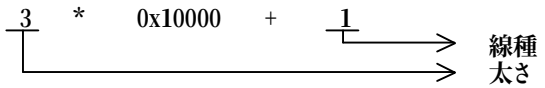
モード	値	説　　明
絶対座標	KLORG	パネル上の絶対座標
相対座標	KLPRV	初めの点以後は、前点からの相対座標

ptn　　上位16ビット　・・・　線の太さを示します。
下位16ビット　・・・　線種を示します。

線種のモードを以下に示します。

モード	値	説　　明
実線	0	
破線	1	
点線	2	
一点破線	3	
二点破線	4	

＜例＞　太さ3ドットの破線を表示する時には次の様に指定します。



*col　　ラインの表示色を示します。
(付録1⑤参照)

pc　　座標データ数を示します。ラインモード(単体)指定時は、ペア
座標数を示します。

*pt　　座標データのアドレスを示します。但し、ラインモードにより座標データ
の構造体が異なります。

ラインモード(継続、単体)　構造体名　*Ktptdt
(グラフ)　構造体名　*Ktptgr

座標データ(Ktptdt, Ktptgr)

x　　各頂点の座標(ピクセル値)を示します。
y

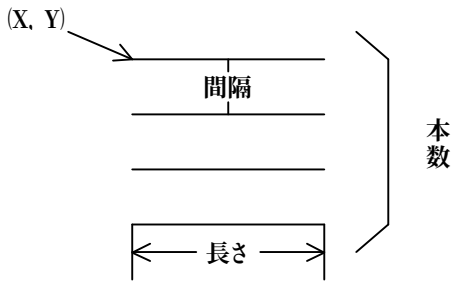
flg(ラインモード　グラフ時のみ)

0　・・・　表示しません。
1　・・・　表示します。
2　・・・　グラフ終端を示します。

罫 線 KOKEI

機 能

罫線の表示を行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         func;       /*  描画ファンクション                  */
    int         mode;       /*  罫線モード                          */
    int         ptn;        /*  太さ/線種                           */
    char        *col;       /*  線色                                */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         len;        /*  長さ                                */
    int         lc;         /*  本数                                */
    int         delta;      /*  間隔                                */
}   Ktkei;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

func　　ラインを描画する際のグラフィック関数を指定します。
通常はGXcopyを指定します。
(付録1⑪参照)

mode　　罫線の表示形態を以下に示します。

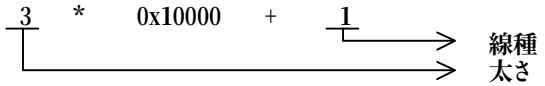
モード	値
横 一定間隔	KKYOKO
縦 一定間隔	KKTATE
横 可変間隔	KKYOKOV
縦 可変間隔	KKTATEV

ptn　　上位16ビット　・・・　線の太さを示します。
　　下位16ビット　・・・　線種を示します。

線種のモードを以下に示します。

モード	値	説　　明
実線	0	――――――――――
破線	1	― ― ― ― ― ― ―
点線	2	‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒
一点破線	3	― ‒ ― ‒ ― ‒ ― ‒ ―
二点破線	4	― ‒ ‒ ― ‒ ‒ ― ‒ ‒ ―

＜例＞　太さ3ドットの破線を表示する時には次の様に指定します。



*col　　罫線の表示色を示します。
　　(付録1⑤参照)

x　　罫線の表示開始位置を示します。
y

len　　罫線の長さを示します。

lc　　罫線の表示本数を示します。

delta　　罫線の間隔を示します。
　　間隔が可変の場合は、間隔を示すデータテーブルアドレスとなります。

(可変の場合のみ間隔データテーブル)

(int) 間隔値	↑
(int) 間隔値	本数-1本
(int) 間隔値	↓

＜例＞　間隔が15, 30, 45, 60の5本線の罫線を表示する時は
　　次の様に指定します。

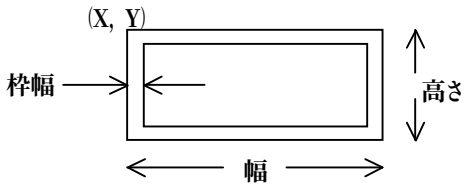
```
int     delta[]   =  {15,30,45,60};

Ktkei     kei    =
{   KOKEI,0,1,GXcopy,KKYOKOV,0,BLACK,
    200,200,150,5,(int)delta
};
```

矩 形 KORECT

機 能

矩形の表示を行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  矩形モード                          */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         w;          /*  幅                                  */
    int         h;          /*  高さ                                */
    int         wh;         /*  枠幅                                */
    char        *c1;        /*  矩形表示色1                         */
    char        *c2;        /*  矩形表示色2                         */
    char        *c3;        /*  矩形表示色3                         */
}   Ktrect;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

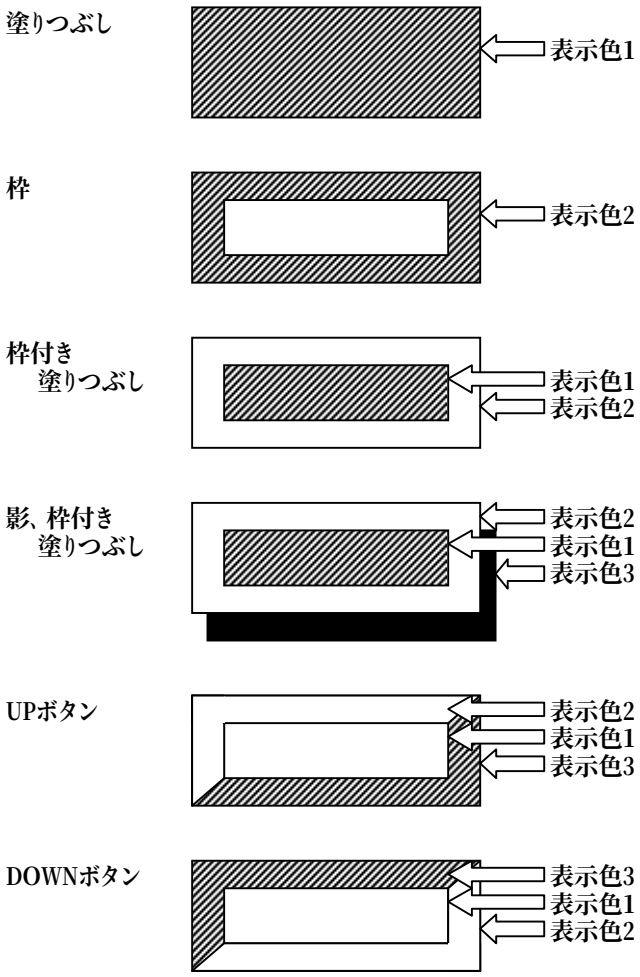
mode　　矩形の表示形態を以下に示します。

モード	値
塗りつぶし	KRBAR
枠	KRWAK
枠付き塗りつぶし	KRBOX
影、枠付き塗りつぶし	KRFLW
UPボタン	KRUPB
DOWNボタン	KRDWB

x
y　　矩形の表示座標位置を示します。

w
h
wh　　矩形の幅(w)、高さ(h)、枠幅(wh)を示します。

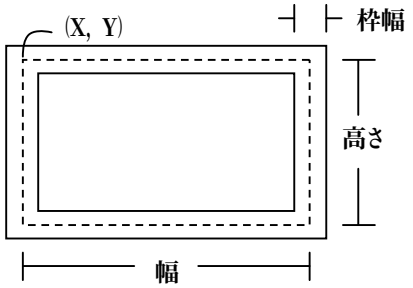
*c1　　矩形の表示色を示します。
*c2　　矩形の表示色1, 2, 3を矩形モードにより以下に示します。
*c3　　(付録1⑤参照)



新矩形　　KONRECT

機　能

矩形の表示を行ないます。
従来の KORECT と違い、立体的な描画は行ないませんが、枠の線種、塗りつぶしパターン等の機能が追加されています。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                      */
    int         mode;       /*  矩形モード                              */
    int         func;       /*  描画ファンクション                      */
    int         fw;         /*  枠幅                                    */
    int         dash;       /*  枠線種                                  */
    int         fmode;      /*  枠描画モード                            */
    char        *ff;        /*  枠フォア表示色                          */
    char        *fb;        /*  枠バック表示色                          */
    int         tile;       /*  塗りつぶしタイル指定                    */
    int         ptn;        /*  固定タイルパターン                      */
    char        *bit;       /*  ユーザタイルパターンデータ              */
    int         bw;         /*  ユーザタイルパターン bitmap 幅          */
    int         bh;         /*  ユーザタイルパターン bitmap 高さ        */
    int         pmode;      /*  塗りつぶしモード                        */
    char        *pf;        /*  塗りつぶしフォア表示色                  */
    char        *pb;        /*  塗りつぶしバック表示色                  */
    int         x;          /*  表示位置  X                             */
    int         y;          /*  表示位置  Y                             */
    int         w;          /*  大きさ  幅                              */
    int         h;          /*  大きさ  高さ                            */
}   Ktnrect;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　矩形の表示形態を以下に示します。

モード	値
枠	KPF
塗りつぶし	KPP
枠付き塗りつぶし	KPF | KPP

func　　通常は GXcopy を指定します。
(付録1⑪参照)

fw　　枠幅は座標線を中心に均等に肉付けされます。但し、枠幅が遇数値の場合には端数分が内側に肉付けされます。
枠幅 0 は、最も高速な描画を行ないます。

dash　　枠には次の線種を使用出来ます。

モード	値	説　　明
実線	KDSOLID	――――――――――
破線	KDDASH	― ― ― ― ― ― ―
点線	KDDOTED	‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒
一点破線	KDSDOT	― ‒ ― ‒ ― ‒ ― ‒ ―
二点破線	KDDDOT	― ‒ ‒ ― ‒ ‒ ― ‒ ‒ ―

fmode　　枠線の描画を行なう際の描画の方法を示します。

モード	値	例
通常	KPNORMAL	
反転	KPREVERSE	
フォアのみ	KPFORE	

 は ff　(フォア色)
 は fb　(バック色)

*ff　　枠表示色を示します。　(付録1⑤参照)
*fb

tile　　塗りつぶし方法を示します。

モード	値
ベタ塗り	KTSOLID
固定タイルパターン	KTFIX
ユーザタイルパターン	KTUSR

ptn　　塗りつぶしタイルのパターン番号を示します。
tile の値が KTFIX の際に有効となります。

0	1	2	3	4	5	6	7
							

*bit　　ユーザが bitmap エディタにて作成したタイルパターンを指定します。
パターンデータは予めソースファイルに include しておきます。

bw　　ユーザが作成したタイルパターンのサイズ (bw:幅、bh:高さ) を
bh　　指定します。正しい値を設定しなければパターンが崩れます。

pmode　　塗りつぶしを行なう際の描画方法を指定します。

モード	値	説　明
通常	KPNORMAL	ベタ指定:フォアの色で描画 タイル指定:パターンをフォア、背景をバックで描画
反転	KPREVERSE	ベタ指定:バックの色で描画 タイル指定:パターンをバック、背景をフォアで描画
フォアのみ	KPFORE	ベタ指定:フォアの色で描画 タイル指定:背景は描画しません

*pf　　塗りつぶし表示色 (pf:フォア、pb:バック) を指定します。
*pb　　(付録1⑤参照)

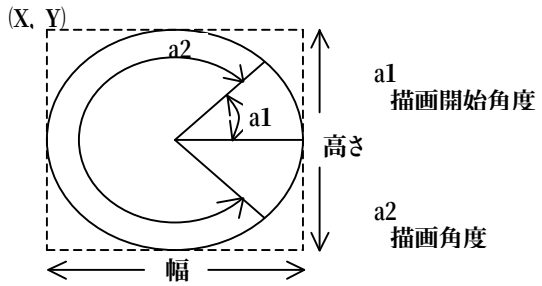
x　　矩形の表示座標位置を示します。
y

w　　矩形の幅 (w) 、高さ (h) を示します。
h

円 弧　　KOARC

機　能

円弧の表示を行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  円弧モード                          */
    int         type;       /*  塗りつぶしタイプ                    */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         w;          /*  円弧 幅                             */
    int         h;          /*  円弧 高さ                           */
    int         as;         /*  描画開始角度                        */
    int         ae;         /*  描画角度                            */
    char        *c1;        /*  塗りつぶし色                        */
    char        *c2;        /*  枠色                                */
}   Ktarc;
```

メンバー説明

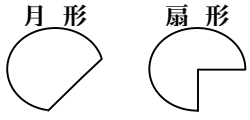
hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　円弧の表示形態を以下に示します。

モード	値	説　　明
塗りつぶし	KAPNT	円弧内を指定色で表示
枠	KAWAK	枠のみを指定色で表示
枠付き塗りつぶし	KACMB	枠と円弧内を指定色で表示

type　　塗りつぶしタイプを以下に示します。

モード	値
月形	KCHOR
扇形	KSLIC



x
y　　円弧の表示座標位置を示します。

w
h　　円弧を囲む長方形の幅、高さを示します。

as
ae　　円弧を描画する角度を示します (0～360)

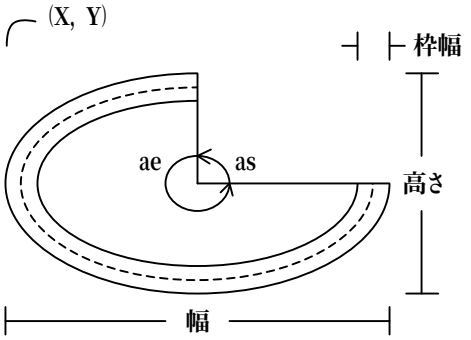
*c1
*c2　　円弧の表示色を示します。
(付録1⑤参照)

MEMO

新円弧　　KONARC

機　能

　円弧の表示を行ないます。
従来の KOARC と違い、枠の
太さ、線種、塗りつぶしパターン
等の機能が追加されています。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  円弧モード                          */
    int         func;       /*  描画ファンクション                  */
    int         fw;         /*  枠幅                                */
    int         dash;       /*  枠線種                              */
    int         fmode;      /*  枠描画モード                        */
    char        *ff;        /*  枠フォア表示色                      */
    char        *fb;        /*  枠バック表示色                      */
    int         tile;       /*  塗りつぶしタイル指定                */
    int         ptn;        /*  固定タイルパターン                  */
    char        *bit;       /*  ユーザタイルパターンデータ          */
    int         bw;         /*  ユーザタイルパターン bitmap 幅      */
    int         bh;         /*  ユーザタイルパターン bitmap 高さ    */
    int         pmode;      /*  塗りつぶしモード                    */
    char        *pf;        /*  塗りつぶしフォア表示色              */
    char        *pb;        /*  塗りつぶしバック表示色              */
    int         amode;      /*  円モード                            */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         w;          /*  大きさ  幅                          */
    int         h;          /*  大きさ  高さ                        */
    int         as;         /*  描画開始角度                        */
    int         ae;         /*  描画角度                            */
}   Ktnarc;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　円弧の表示形態を以下に示します。

モード	値
枠	KPF
塗りつぶし	KPP
枠付き塗りつぶし	KPF | KPP

func　　通常は GXcopy を指定します。
(付録1⑪参照)

fw　　枠幅は座標線を中心に均等に肉付けされます。但し、枠幅が遇数値の
場合には端数分が内側に肉付けされます。
枠幅 0 は、最も高速な描画を行ないます。

dash　　枠には次の線種を使用出来ます。

モード	値	説　　明
実線	KDSOLID	――――――――
破線	KDDASH	― ― ― ― ― ― ―
点線	KDDOTED	‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒
一点破線	KDSDOT	― ‒ ― ‒ ― ‒ ― ‒ ―
二点破線	KDDDOT	― ‒ ‒ ― ‒ ‒ ― ‒ ‒ ―

fmode　　枠線の描画を行なう際の描画の方法を示します。

モード	値	例
通常	KPNORMAL	
反転	KPREVERSE	
フォアのみ	KPFORE	

 は ff　(フォア色)
 は fb　(バック色)

*ff　　枠表示色を示します。　(付録1⑤参照)
*fb

tile　　塗りつぶし方法を示します。

モード	値
ベタ塗り	KTSOLID
固定タイルパターン	KTFIX
ユーザタイルパターン	KTUSR

ptn　　塗りつぶしタイルのパターン番号を示します。
tile の値が KTFIX の際に有効となります。

0	1	2	3	4	5	6	7
							

*bit　　ユーザが bitmap エディタにて作成したタイルパターンを指定します。
パターンデータは予めソースファイルに include しておきます。

bw　　ユーザが作成したタイルパターンのサイズ (bw:幅、bh:高さ) を
bh　　指定します。正しい値を設定しなければパターンが崩れます。

pmode　　塗りつぶしを行なう際の描画方法を指定します。

モード	値	説　明
通常	KPNORMAL	ベタ指定:フォアの色で描画 タイル指定:パターンをフォア、背景をバックで描画
反転	KPREVERSE	ベタ指定:バックの色で描画 タイル指定:パターンをバック、背景をフォアで描画
フォアのみ	KPFORE	ベタ指定:フォアの色で描画 タイル指定:背景は描画しません

*pf　　塗りつぶし表示色 (pf:フォア、pb:バック) を指定します。
*pb　　(付録1⑤参照)

amode　　描画する円弧の形態を指定します。

モード	値
円形	KCHOR
扇形	KSLIC

x　　円弧の表示座標位置を示します。
y

w　　円弧の幅(w)、高さ(h)を示します。
h

as　　円弧の描画開始角度（as）と描画角度（ae）を指定します。
ae　　角度はX軸方向を 0 度とし、反時計まわりで計算します。

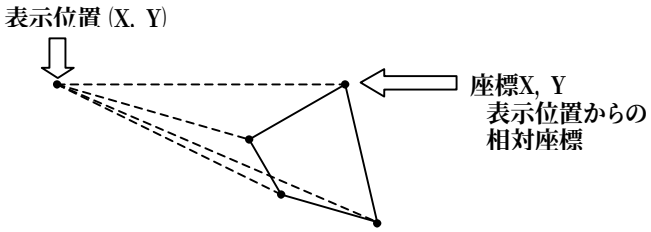
注 意　　タイルパターンでの表示時、御使用の端末によって正しく表示されない場合があります。

MEMO

ポリゴン　KOPOLY

機　能

ポリゴン（多角形）の表示を行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  ポリゴンモード                      */
    int         func;       /*  描画ファンクション                  */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         pc;         /*  座標数                              */
    Ktptdt      *pt;        /*  座標データアドレス                  */
    char        *c1;        /*  塗りつぶし色                        */
    char        *c2;        /*  枠色                                */
}   Ktpoly;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
（2.1部品共通情報参照）

mode　　ポリゴンの表示形態を以下に示します。

モード	値
塗りつぶし	KPPNT
枠	KPWAK
枠付き塗りつぶし	KPCMB

func　　ラインを描画する際のグラフィック関数を指定します。
通常はGXcopyを指定します。
（付録1⑪参照）

x　　ポリゴンの表示座標位置を示します。
y

pc　　ポリゴンの頂点の数を示します。

*pt　　ポリゴン各頂点の座標データアドレスを示します。
（構造体はラインの座標データ構造体参照）

x　　表示位置を原点とした相対座標を示します。
y

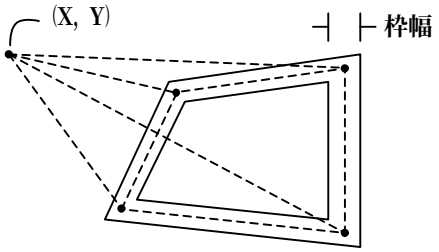
*c1　　ポリゴンの表示色を示します。
*c2　　（付録1⑤参照）

MEMO

新ポリゴン　KONPOLY

機　能

ポリゴン(多角形)の表示を行ないます。
従来の KOPOLY と違い、線の
太さ、線種、塗りつぶしパターン
等の機能が追加されています。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  ポリゴンモード                      */
    int         func;       /*  描画ファンクション                  */
    int         fw;         /*  枠幅                                */
    int         dash;       /*  枠線種                              */
    int         fmode;      /*  枠描画モード                        */
    char        *ff;        /*  枠フォア表示色                      */
    char        *fb;        /*  枠バック表示色                      */
    int         tile;       /*  塗りつぶしタイル指定                */
    int         ptn;        /*  固定タイルパターン                  */
    char        *bit;       /*  ユーザタイルパターンデータ          */
    int         bw;         /*  ユーザタイルパターン bitmap 幅      */
    int         bh;         /*  ユーザタイルパターン bitmap 高さ    */
    int         pmode;      /*  塗りつぶしモード                    */
    char        *pf;        /*  塗りつぶしフォア表示色              */
    char        *pb;        /*  塗りつぶしバック表示色              */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         pc;         /*  座標数                              */
    Ktptdt      *pt;        /*  座標データ                          */
}   Ktnpoly;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　ポリゴンの表示形態を以下に示します。

モード	値
枠	KPF
塗りつぶし	KPP
枠付き塗りつぶし	KPF | KPP

func　　通常は GXcopy を指定します。
(付録1⑪参照)

fw　　枠幅は座標線を中心に均等に肉付けされます。但し、枠幅が遇数値の
場合には端数分が内側に肉付けされます。
枠幅 0 は、最も高速な描画を行ないます。

dash　　枠には次の線種を使用出来ます。

モード	値	説　　明
実線	KDSOLID	――――――――――
破線	KDDASH	― ― ― ― ― ― ― ―
点線	KDDOTED	‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒
一点破線	KDSDOT	― ‒ ― ‒ ― ‒ ― ‒
二点破線	KDDDOT	― ‒ ‒ ― ‒ ‒ ― ‒ ‒

fmode　　枠線の描画を行なう際の描画の方法を示します。

モード	値	例
通常	KPNORMAL	
反転	KPREVERSE	
フォアのみ	KPFORE	

 は ff　(フォア色)
 は fb　(バック色)

*ff　　枠表示色を示します。　(付録1⑤参照)
*fb

tile　　塗りつぶし方法を示します。

モード	値
ベタ塗り	KTSOLID
固定タイルパターン	KTFIX
ユーザタイルパターン	KTUSR

ptn　　塗りつぶしタイルのパターン番号を示します。
tile の値が KTFIX の際に有効となります。

0	1	2	3	4	5	6	7
							

*bit　　ユーザが bitmap エディタにて作成したタイルパターンを指定します。
パターンデータは予めソースファイルに include しておきます。

bw　　ユーザが作成したタイルパターンのサイズ (bw:幅、bh:高さ) を
bh　　指定します。正しい値を設定しなければパターンが崩れます。

pmode　　塗りつぶしを行なう際の描画方法を指定します。

モード	値	説　明
通常	KPNORMAL	ベタ指定:フォアの色で描画 タイル指定:パターンをフォア、背景をバックで描画
反転	KPREVERSE	ベタ指定:バックの色で描画 タイル指定:パターンをバック、背景をフォアで描画
フォアのみ	KPFORE	ベタ指定:フォアの色で描画 タイル指定:背景は描画しません

*pf　　塗りつぶし表示色 (pf:フォア、pb:バック) を指定します。
*pb　　(付録1⑤参照)

x　　新ポリゴンの表示座標位置 (基準点) を示します。
y

pc　　ポリゴンの頂点の数を示します。

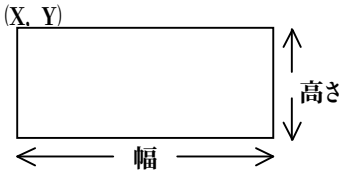
*pt　　表示位置 (x, y) からの相対座標データ。(構造体の詳細はライン部品を参照)

注 意　　タイルパターンでの描画時、御使用の端末によって正しく表示されない場合があります。

ビットマップ　　KOIMAG

機　能

ビットマップ画像、又は、
ラスター画像の表示を行ないます。



(補足)　ラスター画像を表示する際に使用する色資源は、既に同一色が使用されていれば共有して利用します。色数が多い画像の場合には表示に時間が掛かる場合があります。

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  ビットマップモード                  */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         w;          /*  イメージ 幅                         */
    int         h;          /*  イメージ 高さ                       */
    char        *fc;        /*  イメージ 表示色                     */
    char        *bc;        /*  イメージ 背景色                     */
    char        *name;      /*  イメージデータアドレス              */
    int         flg;        /*  データフラグ                        */
    Pixmap      pix;        /*  イメージPixmap                      */
}   Ktimag;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　画像イメージの形式を示します。

モード	値	説　　明
ファイルデータ	KIFIL	ファイルから読み込む
メモリデータ	KIDAT	メモリから読み込む

ファイルデータ指定時は、ディスク上にある画像ファイルを使用します。
メモリデータ指定時は、メモリ上にある画像データを使用します。

x　　画像の表示座標位置を示します。
y

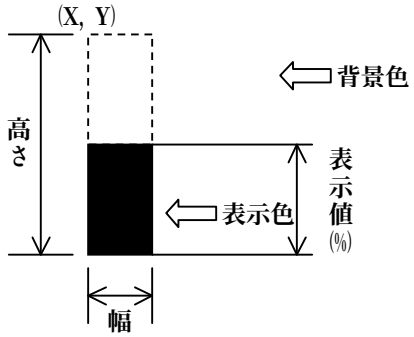
w　　画像を表示するエリアの幅、高さを示します。
h　　modeが「メモリデータ」の際に、画像のフォーマットが bitmap の
場合には、画像データに画像のサイズ情報が入っていませんので、
実際の画像サイズを設定する必要があります。

*fc *bc	ビットマップの表示色、背景色を示します。 （付録1⑤参照）
*name	ファイルデータ指定時は、画像を保存しているファイルパスを設定します。 メモリデータ指定時は、メモリ上に格納している画像データのアドレスを設定します。
flg	ライブラリで管理する状態フラグです。初期値は0を指定します。
pix	ライブラリで管理するイメージ情報です。初期値は0を指定します。

棒グラフ　KOBARG

機　能

棒グラフの表示を行ないます。
棒グラフの長さを変更する場合には、グラフ表示値(val)の値を変更し、TKdisplayを行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  グラフモード                        */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         w;          /*  グラフ  幅                          */
    int         h;          /*  グラフ  高さ                        */
    char        *fc;        /*  グラフ表示色                        */
    char        *bc;        /*  グラフ表示背景色                    */
    float       val;        /*  グラフ表示値(%)                     */
    int         flg;        /*  フラグ                              */
    int         old;        /*  イベント管理情報                    */
}   Ktbarg;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　グラフ表示方向を以下に示します。

モード	値
縦棒グラフ	KBTATE
横棒グラフ	KBYOKO

x　　表示座標位置を示します。
y

w　　表示するグラフの幅、高さを示します。
h

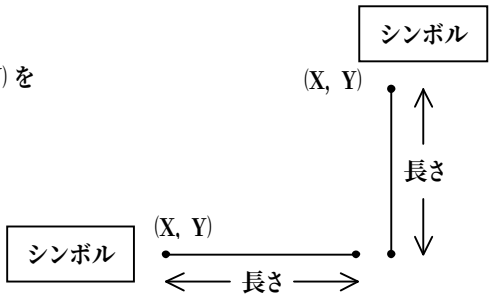
*fc　　グラフ表示色を指定します。
(付録1⑤参照)

*bc	グラフを表示しようとする領域の色を設定します。 この設定が誤っていると、グラフ表示色が設定した色になりません。 （付録1⑤参照）
val	グラフ表示の割合（%）を示します。（0.0～100.0%）
flg	ライブラリで管理する為、初期値は0を指定します。
old	ライブラリで管理する為、初期値は0を指定します。

バーカーソル　　KOBARC

機　能

バーカーソルの表示を行ないます。
表示位置を変更する際には、位置(X, Y)を
変更しTKdisplayを行なうだけです。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  バーカーソルモード                  */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         l;          /*  カーソル  長さ                      */
    char        *col;       /*  表示色                              */
    int         font;       /*  シンボルフォントタイプ              */
    char        *sym;       /*  シンボルストリングスアドレス        */
    int         flg;        /*  状態フラグ                          */
    int         ox;         /*  前表示位置  X                       */
    int         oy;         /*  前表示位置  Y                       */
}   Ktbarc;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　バーカーソルの表示形態を以下に示します。

モード	値
縦	KBTATE
横	KBYOKO

x　　バーカーソルの表示座標位置を示します。
y

l　　カーソルの長さを示します。

*col　　バーカーソルの表示色を示します。
(付録1⑤参照)

font　　バーカーソルのシンボルで使用するフォントのコードを示します。
(付録1④-1参照)

*sym　　バーカーソルのシンボルテキストのストリングスアドレスを示します。

flg　　バーカーソルの表示状態を示すフラグでライブラリで管理します。
初期値は0を指定します。

ox　　バーカーソルの前回の表示位置を示し、ライブラリで管理します。
oy　　初期値は0を指定します。

MEMO

ボックスカーソル　KOBOXC

機　能

ボックスカーソルの表示を行ないます。ボックスカーソルとは、図形を描画する際に、表示色を排他演算(xor)を行ないます。これにより、パネル上の元図形を破壊することなく図形を重ねて描画されます。

ボックスカーソルを移動する際には、位置(X, Y)及び、大きさ(w, h)を変更し、TKdisplayを行ないます。

枠　　塗りつぶし　　円

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  ボックスカーソルモード              */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         w;          /*  幅                                  */
    int         h;          /*  高さ                                */
    char        *col;       /*  表示色                              */
    char        *bak;       /*  背景色                              */
    int         flg;        /*  状態フラグ                          */
    int         ox;         /*  前表示位置  X                       */
    int         oy;         /*  前表示位置  Y                       */
    int         ow;         /*  前幅                                */
    int         oh;         /*  前高さ                              */
}   Ktboxc;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　ボックスカーソルの表示モードを以下に示します。

モード	値
枠のみ	KCBOX
塗りつぶし	KCREC
円	KCARC

x
y　　ボックスカーソルの表示座標位置を示します。

w
h　　ボックスカーソルの幅、高さを示します。

*col　　ボックスカーソルの表示色を示します。
(付録1⑤参照)

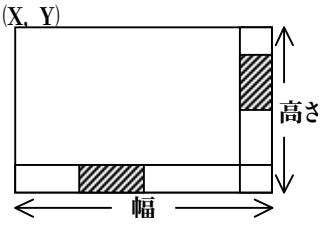
*bak	ボックスカーソルの背景色を示します。 （付録1⑤参照）
flg	ボックスカーソルの表示状態を示すフラグです。ライブラリで管理します。 初期値は0を指定します。
ox oy	ボックスカーソルの前回表示位置です。ライブラリで管理します。 初期値は0を指定します。
ow oh	ボックスカーソルの前回の幅、高さを示します。ライブラリで管理します。 初期値は0を指定します。

ラスター　　KORAST

機　能

　SUNラスターフォーマットの画像イメージを表示します。
　24bitイメージデータ(フルカラー)の画像を使用する際には、フルカラー表示可能な環境でご利用する事をお勧めします。

(注)圧縮フォーマットには対応していません。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  ラスターモード                      */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         w;          /*  イメージ 幅                         */
    int         h;          /*  イメージ 高さ                       */
    char        *name;      /*  イメージファイル名                  */
    int         flg;        /*  データフラグ                        */
    int         *info;      /*  インフォメーションデータ            */
    QID         event;      /*  イベント処理情報                    */
}   Ktrast;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　ラスターモードの形式を以下に示します。

モード	値
スクロールバーなし	KRNORM
スクロールバーあり	KRSBAR

スクロールバーありに指定していても実際の画像データが、指定した表示エリアの大きさより小さい場合は、スクロールバーは表示されません。
スクロールバーは、幅、高さで指定したエリアの内側に表示されます。

x
y　　画像の表示座標位置を示します。

w
h　　画像を表示するエリアの幅、高さを示します。

*name　　画像ファイルのパス名称のアドレスを設定します。

flg　　ライブラリで管理する状態フラグです。初期値は0を指定します。

*info　　ライブラリで管理する情報です。初期値は0を指定します。

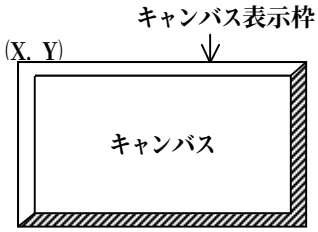
event　　ライブラリで管理する情報です。初期値は0を指定します。

MEMO

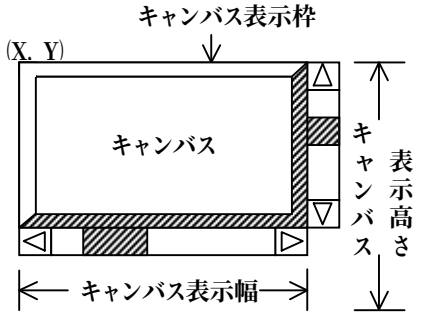
キャンバス　KOCANV

機　能

キャンバスの表示を行ないます。
キャンバスとは、サブパネルとよく似たパネルの一種です。



スクロールバーなし



スクロールバーあり

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  キャンバス表示モード                */
    int         x;          /*  キャンバス表示枠 表示位置  X        */
    int         y;          /*  キャンバス表示枠 表示位置  Y        */
    int         w;          /*  キャンバス表示枠 幅                 */
    int         h;          /*  キャンバス表示枠 高さ               */
    int         wh;         /*  キャンバス表示枠 枠幅               */
    int         cw;         /*  キャンバス 幅                       */
    int         ch;         /*  キャンバス 高さ                     */
    int         dx;         /*  キャンバス 表示位置  X              */
    int         dy;         /*  キャンバス 表示位置  Y              */
    char        *cc;        /*  キャンバス 背景色                   */
    char        *pc;        /*  パネル色                            */
    char        *hc;        /*  ハイライト明色                      */
    char        *bc;        /*  ハイライト暗色                      */
    int         *inf;       /*  部品管理情報                        */
}   Ktcanv;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　キャンバスの表示形式を以下に示します。

モード	値
スクロールバーなし	KCRNORM
スクロールバーあり	KCSBAR

x
y　　キャンバス表示枠の表示位置を示します。

w h	キャンバス表示枠の幅、高さを示します。
wh	キャンバス表示枠の枠幅を示します。
cw ch	キャンバスの幅、高さを示します。
dx dy	キャンバスの表示位置を示します。
*cc	キャンバスの背景色を示します。 （付録1⑤参照）
*pc	キャンバス表示枠の内部表示色、スクロールバー表示色を示します。 （付録1⑤参照）
*hc *bc	ハイライト表示色を示します。 （付録1⑤参照）
*inf	ライブラリが部品を管理する為の情報。 初期値は0を指定します。

・キャンバスに部品／図形を描画する方法

キャンバス部分は実際には、サブパネルになっている為、この部分に部品を描画する為には、core情報が必要です。
キャンバス部分のcore情報を取得する関数TKcanvcoreを使用する事で実現します。これで取得したcore情報を使用して、通常のパネルに描画する手順で部品を登録します。

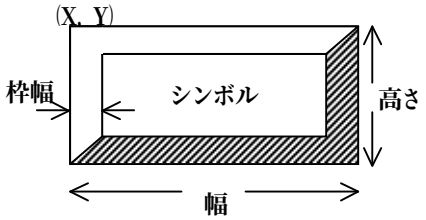
注 意

TKcanvcoreを実行する際には、必ずキャンバス部品は登録されていなければいけません。

ボタン　　KOBUTN

機　能

ボタンイメージの表示とイベント処理を行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;          /*  オブジェクトヘッダ                  */
    int         mode;         /*  ボタンモード                        */
    int         x;            /*  表示位置 X                          */
    int         y;            /*  表示位置 Y                          */
    int         w;            /*  幅                                  */
    int         h;            /*  高さ                                */
    int         wh;           /*  枠幅                                */
    int         font;         /*  シンボルフォントコード              */
    char        *sym;         /*  シンボルストリングス                */
    char        *pc1;         /*  ボタンパネル色1                     */
    char        *pc2;         /*  ボタンパネル色2                     */
    char        *hc1;         /*  ボタンハイライト色1                 */
    char        *hc2;         /*  ボタンハイライト色2                 */
    char        *sc1;         /*  シンボル表示色1                     */
    char        *sc2;         /*  シンボル表示色2                     */
    XID         bid;          /*  ボタンID                            */
    int         (*subon) ();  /*  コールバックルーチン1               */
    int         (*suboff) (); /*  コールバックルーチン2               */
    int         flg;          /*  ボタン状態                          */
    Pixmap      pix;          /*  イメージPixmap                      */
    QID         event;        /*  イベント処理情報                    */
}   Ktbutn;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　ボタンの形式を以下に示します。

モード	値
モーメンタリボタン	KBMOM
オルタネートボタン	KBALT

【KBMOM】 モーメンタリボタンはマウスのクリックに連動し、クリックされた時にコールバックルーチン1が呼ばれ、コールバックルーチン2は呼ばれません。

【KBALT】 オルタネートボタンはマウスのクリックでDOWN/UPの状態を交互に繰り返し、DOWN時にコールバックルーチン1が、UP時にコールバックルーチン2が呼ばれます。

x　　ボタンの表示座標位置を示します。
y

w　　ボタンの幅、高さを示します。
h

wh　　ボタンの枠幅を示します。

font　　シンボルのフォントタイプを指定します。(付録1④-1, ④-2参照)

*sym　　fontメンバーの指定により設定値の内容が変わります。

文字データ種別を指定した場合
- ボタンに表示する文字列を指定します。

ビットマップデータ種別を指定した場合
- ファイルデータでは、画像ファイルのパスを指定します。

- メモリデータでは、画像データを格納しているメモリアドレスを指定します。(ビットマップデータの場合には、ボタン枠内のサイズと画像のサイズが同じでなければなりません。)

※ 利用可能な画像フォーマットは、「ビットマップ」「SUNラスター」です。

*pc1　　ボタンのパネル表示色の色コードを示します。
*pc2

パネル色1　・・・　ボタンUP時(解放)の表示色。
パネル色2　・・・　ボタンDOWN時(押し下げ)の表示色。

*hc1　　ボタン枠の表示色を示します。(付録1⑤参照)
*hc2

ハイライト色1　・・・　明部表示色。
ハイライト色2　・・・　暗部表示色。

*sc1　　ボタンシンボルの表示色を示します。(付録1⑤参照)
*sc2

シンボル表示色1　・・・　ボタンUP時(解放)の表示色。
シンボル表示色2　・・・　ボタンDOWN時(押し下げ)の表示色。

bid　　ボタン識別ユーザ定義IDを示します。

*subon　　ボタン操作時に呼ばれるサブルーチンを指定します。
*suboff　　マウスボタンの プレス/リリース 各々で呼ばれます。

subon　・・・　ボタン押し下げ時に呼び出される。
suboff　・・・　ボタン押し上げ時に呼び出される。

注 意　ボタンモードがモーメンタリボタンであれば、コールバックルーチン2は呼ばれません。

flg　　ボタンの状態を示します。

OFF　・・・　UP(押し上げ)状態を示す。
ON　・・・　DOWN(押し下げ)状態を示す。
-1　・・・　ボタンが半透明で表示され、イベントを受け付けません。

pix　　ボタンシンボルでビットマップを使用した場合のビットマップPixmapデータをセットします。
ライブラリで管理する為、初期値は0を指定します。

event　　ボタン部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

・ ボタン処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, butn, event);

    Ktcore      *core;      /*    パネル管理情報                    */
    Ktbutn      *butn;      /*    イベント発生ボタン部品            */
    int         event;      /*    イベントタイプ                    */
                            ->マウスボタン押し下げ ・・・・・・ ButtonPress
                              マウスボタン解放     ・・・・・・ ButtonRelease
```

リターンステータス

```
    0              ・・・  正常終了
    0 より小さい   ・・・  次オブジェクトチェック
    0 より大きい   ・・・  イベント処理終了
```

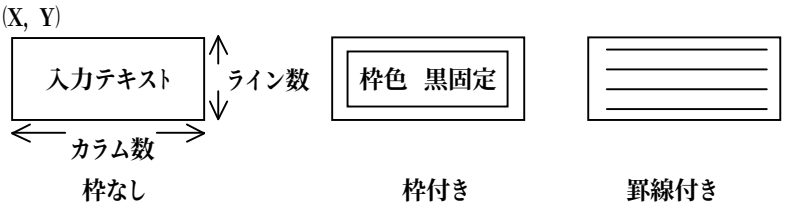
注 意　マウスボタンイベントによりコールバックルーチンは二度呼ばれる為、コールバックルーチン内にてマウスボタンのイベントタイプを必ずチェックして下さい。

MEMO

テキスト入力　　KOITEXT

機　能

複数行にわたるテキストデータ入力表示とイベント処理を行ないます。



枠なし　　枠付き　　罫線付き

入力状態の終了方法

・ ライン数1行の時　・・・・・・ ESC キー　又は Return キー

・ ライン数複数行の時　・・・ ESC キー

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  入力モード                          */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         c;          /*  カラム数                            */
    int         l;          /*  ライン数                            */
    char        *fc;        /*  テキスト表示色                      */
    char        *bc;        /*  背景色                              */
    int         font;       /*  使用フォントコード                  */
    char        *text;      /*  入力文字バッファ                    */
    XID         id;         /*  テキスト入力  ID                    */
    int         (*subb) (); /*  前コールバックルーチン              */
    int         (*sube) (); /*  後コールバックルーチン              */
    int         flg;        /*  状態フラグ                          */
    QID         event;      /*  イベント処理情報                    */
}   Ktitext;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　テキスト入力の表示形式を以下に示します。

モード	値
テキスト入力枠なし	KINOM
テキスト入力枠付き	KIWAK
テキスト入力罫線表示	KIKEI

x
y　　テキスト入力の表示位置を示します。

c　　テキスト入力エリアの半角での文字数を示します。

l	テキスト入力エリアの行数を示します。
*fc *bc	表示を行なうテキストの文字色と背景色を示します。 (付録1⑤参照)
font	テキスト入力文字の表示フォントコードを示します。 (付録1④-1参照)
*text	入力文字を格納するバッファを示します。
id	テキスト入力識別ユーザ定義IDを示します。
*subb *sube	キー入力開始前、及び終了時に呼び出されるサブルーチンを指定します。
flg	テキスト入力の状態を示します。 キー入力　OFF ･･･ KIOFF キー入力　ON　･･･ KION
event	テキスト入力部品のイベント処理管理情報のQIDを示します。 ライブラリにて管理する為、初期値は0を指定します。

・ テキスト入力処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名 (core, itext) ;

    Ktcore      *core;      /*    パネル管理情報          */
    Ktitext     *itext;     /*    テキスト入力部品        */
```

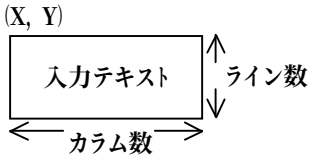
リターンステータス

0	･･･	正常終了
0 より小さい	･･･	前処理の場合:選択解除 後処理の場合:正常終了
0 より大きい	･･･	イベント処理終了

新テキスト入力　　KONTEXT

機　能

複数行にわたるテキストデータの入力/表示とイベント処理を行ないます。
また、テキストデータが表示エリアをはみだす時は、スクロールバーを表示させる事が出来ます。
このスクロールバーのスライダを移動させると、スライダの動きに合わせてテキスト表示部分も移動し、テキストデータ全体を見る事が出来ます。



入力状態の終了方法
- ライン数1行の時　・・・・・・ ESC キー　又は Return キー
- ライン数複数行の時　・・・ ESC キー

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  モード                              */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         c;          /*  カラム数                            */
    int         l;          /*  ライン数                            */
    int         mc;         /*  最大カラム数                        */
    int         ml;         /*  最大ライン数                        */
    int         font;       /*  使用フォントコード                  */
    char        *text;      /*  入力文字バッファ                    */
    int         size;       /*  バッファサイズ                      */
    char        *fc;        /*  テキスト表示色                      */
    char        *bc;        /*  背景色                              */
    char        *pc;        /*  スクロールバー色                    */
    char        *hc;        /*  ハイライト明色                      */
    char        *sc;        /*  ハイライト暗色                      */
    XID         id;         /*  新テキスト ID                       */
    int         (*subb) ();  /*  前コールバックルーチン             */
    int         (*sube) ();  /*  後コールバックルーチン             */
    int         flg;        /*  状態フラグ                          */
    int         *inf        /*  部品管理情報                        */
    QID         event;      /*  イベント処理情報                    */
}   Ktntext;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

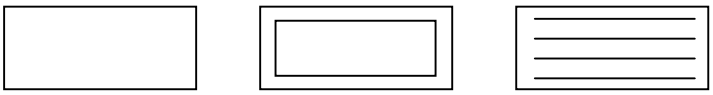
mode　　入力モードと表示モードの OR をとり設定します。

・ 入力モード

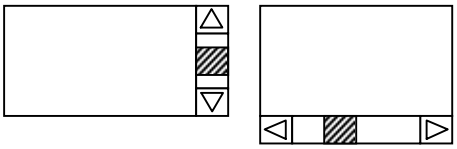
モード	値
入力あり	KION
表示のみ	KIOFF

・ 表示モード

モード	値
枠付き	KITW
罫線付き	KITK
縦スクロール･バー付き	KITVS
横スクロール･バー付き	KITHS



枠なし　　枠付き　　罫線付き



縦スクロールバー付き　横スクロールバー付き

注 意

枠なしモードで表示させる場合は、入力モードのみを設定します。
入力モードが設定されていない場合は、入力なしとなります。
表示モードは複数のモードを OR により組み合わせる事が出来ます。

例: 入力ありで罫線、縦スクロールバー付きの設定は、
KION |KITK |KITVS
となります。

x　　新テキストの表示位置を示します。
y

c　　新テキスト表示エリアの半角での文字数を示します。

l　　新テキスト表示エリアの行数を示します。

mc　　新テキスト表示エリアをはみだして表示する場合、スクロールバーによって表示する事の出来る最大文字数を半角で示します。

ml　　新テキスト表示エリアをはみだして表示する場合、スクロールバーによって表示する事の出来る最大行数を示します。

font　　新テキスト文字の表示フォントコードを示します。
(付録1④-1参照)

*text　　入力文字を格納するバッファを示します。

size　　入力文字を格納するバッファのサイズを示します。

*fc　　表示を行なうテキストの文字色と背景色を示します。
*bc　　(付録1⑤参照)

*pc　　スクロールバー色を示します。
(付録1⑤参照)

*hc　　枠、スクロールバー等のハイライト表示色を示します。
*sc　　(付録1⑤参照)

id　　新テキスト識別ユーザ定義IDを示します。

*subb　　キー入力開始前、及び終了時に呼び出されるサブルーチンを指定します。
*sube

flg　　新テキストの状態を示します。

キー入力　OFF ・・・ KIOFF
キー入力　ON　・・・ KION

*inf　　ライブラリが部品を管理する為の情報。
初期値は0を指定します。

event　　新テキスト部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

・ 新テキスト処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名 (core, ntext) ;

    Ktcore      *core;      /*    パネル管理情報          */
    Ktntext     *ntext;     /*    新テキスト部品          */
```

リターンステータス

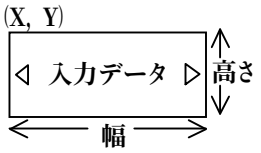
0　　・・・　正常終了
0 より小さい　・・・　前処理の場合:選択解除
　　後処理の場合:正常終了
0 より大きい　・・・　イベント処理終了

MEMO

データ入力　　KOIDATA

機　能

一行のデータ表示と入力イベント処理を行ないます。
また、入力データが表示エリアをはみだす時は、ボタンが表示されます。
このボタンをクリックする事により、入力データ全体を見る事が出来ます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  モード                              */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         w;          /*  幅                                  */
    int         h;          /*  高さ                                */
    int         c;          /*  データ表示文字数                    */
    int         font;       /*  使用フォントコード                  */
    char        *fc;        /*  テキスト表示色                      */
    char        *bc;        /*  背景色                              */
    char        *hc;        /*  ハイライト明色                      */
    char        *sc;        /*  ハイライト暗色                      */
    int         posi;       /*  データ表示位置                      */
    int         type;       /*  データタイプ                        */
    int         prec;       /*  小数点以下有効桁数                  */
    int         *data;      /*  入力文字バッファ                    */
    XID         id;         /*  データ入力 ID                       */
    int         (*subb) (); /*  前コールバックルーチン              */
    int         (*sube) (); /*  後コールバックルーチン              */
    int         flg;        /*  状態フラグ                          */
    int         *inf        /*  部品管理情報                        */
    QID         event;      /*  イベント処理情報                    */
}   Ktidata;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

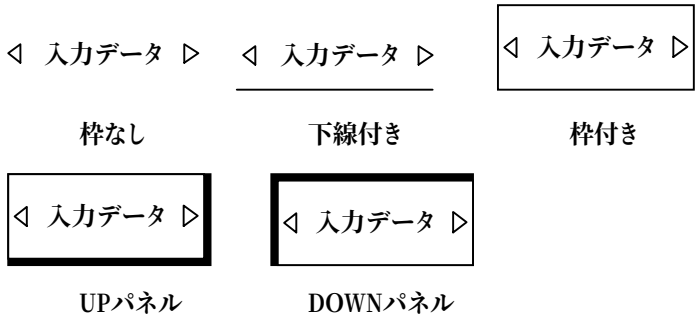
mode　　入力モードと表示モードの OR をとり設定します。

・ 入力モード

モード	値
入力あり	KION
表示のみ	KIOFF

・ 表示モード

モード	値
下線付き	KIDINL
枠付き	KIDINR
UPパネル	KIDINU
DOWNパネル	KIDIND



注 意

枠なしモードで表示させる場合は、入力モードのみを設定します。
入力モードが設定されていない場合は、入力なしとなります。

x
y　　データ入力の表示位置を示します。

w
h　　データ入力の幅、高さを示します。

c　　表示エリア内への表示文字（半角）数を示します。
実際の文字数が表示エリアのサイズを越える場合は、ボタンが
表示されスクロールする事が可能となります。

font　　データ入力文字の表示フォントコードを示します。
（付録1④-1参照）

*fc
*bc　　表示を行なうテキストの文字色と背景色を示します。
（付録1⑤参照）

*hc
*sc　　枠、ボタン等のハイライト表示色を示します。
（付録1⑤参照）

posi　　データ入力エリアに対するデータの表示位置を示します。

表示位置	値
右詰め	KPR
中央	KPM
左詰め	KPL

type　　表示入力を行なうデータの型を示します。

値	表示形式	データ型
KDDEC	10進数	int
KDHEX	16進数	int
KDOCT	8進数	int
KDFLOAT	小数点表示	float
KDEXP	指数表示	float
KDGFORM	上記2つの短い方	float
KDNUM	数字	char
KDANK	アスキー （日本語不可）	char
KDTXT	テキスト （日本語可）	char

prec　　データ型が KDFLOAT, KDEXP, KDGFORM の場合、小数点以下の
有効桁数を指定します。

*data　　入力文字を格納するバッファを示します。

id　　データ入力識別ユーザ定義IDを示します。

*subb　　キー入力開始前、及び終了時に呼び出されるサブルーチンを指定します。
*sube

flg　　データ入力の状態を示します。

キー入力　OFF ・・・ KIOFF
キー入力　ON ・・・ KION

*inf　　ライブラリが部品を管理する為の情報。
初期値は0を指定します。

event　　データ入力部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

・ データ入力処理 前コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, idata, mask);

    Ktcore      *core;      /*    パネル管理情報                          */
    Ktidata     *idata;     /*    データ入力部品                          */
    int         mask;       /*    データ入力選択マウスボタンマスク        */
                    ---->マウス左ボタン ・・・・・・ Button1Mask
                         マウス中ボタン ・・・・・・ Button2Mask
                         マウス右ボタン ・・・・・・ Button3Mask
                         セル移動    ・・・・・・・・・ NULL
```

リターンステータス

```
    0             ・・・  正常終了
    0 より小さい  ・・・  データ入力の選択解除
    0 より大きい  ・・・  イベント処理の終了
```

補足:「表示のみ」の場合でも呼ばれます。入力フィールドの切り換えはありません。

・ データ入力処理 後コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, idata, type, text);

    Ktcore      *core;      /*    パネル管理情報                          */
    Ktidata     *idata;     /*    データ入力部品                          */
    int         type;       /*    終了タイプ                              */
                    ---->KEYESC     ・・・  エスケープ
                         KEYCR      ・・・  リターン
                         KEYTAB     ・・・  タブ
                         KEYBAT     ・・・  シフト + タブ
                         KEYUP      ・・・  ↑
                         KEYDW      ・・・  ↓
                         KEYMUS     ・・・  マウス操作  (環境変数KTlSUB設定時のみ)
                         KEYSW      ・・・  関数操作    (環境変数KTlSUB設定時のみ)
    char        *text;      /*    入力テキスト                            */
```

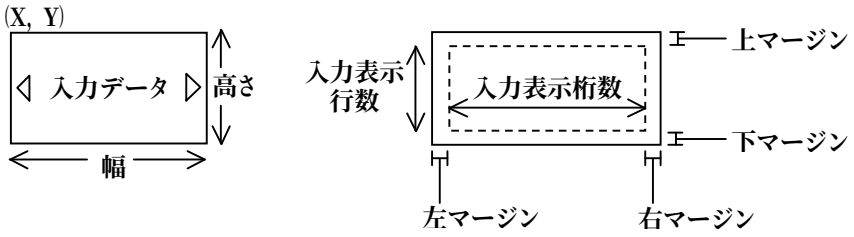
リターンステータス

```
    0             ・・・  正常終了
    0 より小さい  ・・・  データ入力の再入力
    0 より大きい  ・・・  イベント処理の終了
```

データ3　KODATA3

機　能

複数行にわたるデータ入力表示とイベント処理を行ないます。
また、入力表示行が1行で入力データが表示エリアをはみだす時は、
ボタンが表示されます。
このボタンをクリックする事により、入力データ全体を見る事が出来ます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  モード                              */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         w;          /*  幅                                  */
    int         h;          /*  高さ                                */
    int         c;          /*  入力表示桁数                        */
    int         line;       /*  入力表示行数                        */
    int         font;       /*  使用フォントコード                  */
    char        *fc;        /*  テキスト表示色                      */
    char        *bc;        /*  背景色                              */
    char        *hc;        /*  ハイライト明色                      */
    char        *sc;        /*  ハイライト暗色                      */
    int         posi;       /*  左右データ表示位置                  */
    int         posi2;      /*  上下データ表示位置                  */
    int         type;       /*  データタイプ                        */
    int         prec;       /*  小数点以下有効桁数                  */
    int         limit;      /*  リミットチェックモード              */
    double      max;        /*  上限値                              */
    double      min;        /*  下限値                              */
    int         idec;       /*  確定文字数                          */
    int         hmd;        /*  修飾                                */
    int         sp;         /*  横/斜線種 太さ                      */
    char        *hmdc;      /*  網掛け/斜線色                       */
    int         imod;       /*  挿入/上書きモード                   */
    int         cx;         /*  カーソル桁位置                      */
    int         cy;         /*  カーソル行位置                      */
    char        *chk;       /*  入力可能文字                        */
    char        *fmt;       /*  表示フォーマット                    */
    int         *data;      /*  入力データバッファ                  */
    XID         id;         /*  データ3 ID                          */
    int         (*subb) (); /*  前コールバックルーチン              */
    int         (*sube) (); /*  後コールバックルーチン              */
    int         flg;        /*  状態フラグ                          */
    int         *inf        /*  部品管理情報                        */
    QID         event;      /*  イベント処理情報                    */
}   Ktdata3;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　入力モードと表示モードの OR をとり設定します。

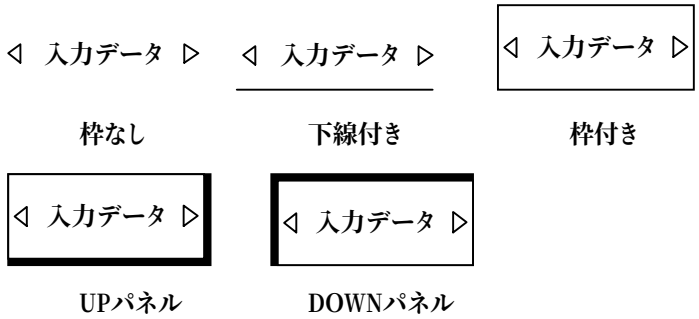
・ 入力モード

モード	値
入力あり	KION
表示のみ	KIOFF
表計算	KIDEXP

← 入力ありとの併用禁止

・ 表示モード

モード	値
下線付き	KIDINL
枠付き	KIDINR
UPパネル	KIDINU
DOWNパネル	KIDIND



注 意

枠なしモードで表示させる場合は、入力モードのみを設定します。
入力モードが設定されていない場合は、入力なしとなります。

x　　データ入力の外枠の表示位置 (左上) を示します。
y

w　　データ入力の外枠の幅、高さを示します。
h

c　　※ 入力表示行数が1の時

表示エリア内への表示文字 (半角) 数を示します。
実際の文字数が表示エリアのサイズを越える場合は、ボタンが
表示されスクロールする事が可能となります。

※ 入力表示行数が2以上の時

表示エリア内への1行分の表示文字 (半角) 数を示します。
入力表示文字数が表示エリアのサイズを越える場合は、表示エリア
のサイズになります。

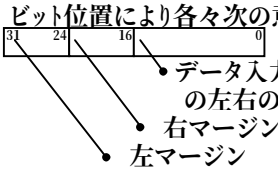
line　　表示エリアの行数を示します。
入力表示行数が表示エリアのサイズを越える場合は、表示エリア
のサイズになります。

font　　データ入力文字の表示フォントコードを示します。
(付録1④-1参照)

*fc　　表示を行なうテキストの文字色と背景色を示します。
*bc　　(付録1⑤参照)

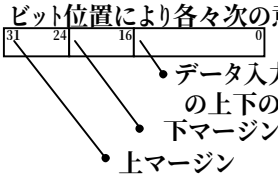
*hc　　枠、ボタン等のハイライト表示色を示します。
*sc　　(付録1⑤参照)

posi　　ビット位置により各々次の意味を持っています。



表示位置	値
右詰め	KPR
中央	KPM
左詰め	KPL

posi2　　ビット位置により各々次の意味を持っています。



表示位置	値
上詰め	KTBUP
中央	KTBMI
下詰め	KTBDW

type　　表示入力を行なうデータの型を示します。

値	表示形式	データ型
KDDEC	10進数	int
KDHEX	16進数	int
KDOCT	8進数	int
KDUDEC	10進数	unsigned int
KDUHEX	16進数	unsigned int
KDUOCT	8進数	unsigned int
KDSDEC	10進数	short
KDSHEX	16進数	short
KDSOCT	8進数	short
KDUSDEC	10進数	unsigned short
KDUSHEX	16進数	unsigned short
KDUSOCT	8進数	unsigned short
KDFLOAT	小数点表示	float
KDEXP	指数表示	float
KDGFORM	上記2つの短い方	float
KDDOUBL	小数点表示	double
KDDEXP	指数表示	double
KDDGFRM	上記2つの短い方	double
KDNUM	数字	char
KDANK	ANK文字　(日本語不可)	char
KDTXT	テキスト　(日本語可)	char
KDPWD	パスワード入力	char

prec
- データタイプが KDFLOAT, KDEXP, KDGFORM, KDDOUBL, KDDEXP, KDDGFRM の時、小数点以下の有効桁数を示します。
- データタイプが KDPWD の時、ANKコード(半角文字)を設定します。
  キー入力の際に、この文字で表示されます。ANKコード以外の場合には、デフォルトの '*' で表示します。

limit　　上下限値(数値)チェックの有効/無効を指定します。

値	説　明
KTBLUP	上限値有効
KTBLDWN	下限値有効
NULL	無効

補 足　　上/下限値共に有効にする場合は、(KTBLUP |KTBLDWN)と設定します。

max　　ここで指定された数値よりも大きい数値は入力出来ません。

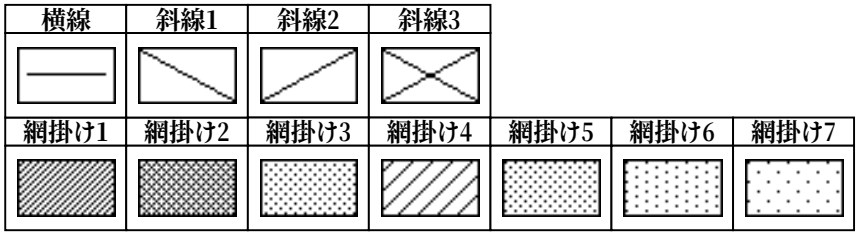
min　　ここで指定された数値よりも小さい数値は入力出来ません。

idec　　入力が指定した文字数に達すると確定となります。
0を設定すると、入力表示桁数 * 入力表示行数で確定となります。
この設定を無効にしたい場合は、-1を設定して下さい。

hmd　　セルの修飾を指定します。

値	モード
KTBNON	なし
KTBREV	反転
KTBLINE	横線
KTBSLA1	斜線1
KTBSLA2	斜線2
KTBSLA3	斜線3
KTBMES1	網掛け1
KTBMES2	網掛け2
KTBMES3	網掛け3
KTBMES4	網掛け4
KTBMES5	網掛け5
KTBMES6	網掛け6
KTBMES7	網掛け7

各モードによる表示イメージを次に示します。

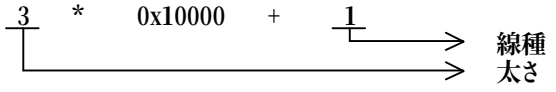


sp　　上位16ビット　・・・　横、斜線の太さを示します。
下位16ビット　・・・　横、斜線種を示します。

線種のモードを以下に示します。

モード	値	説　　明
実線	0	――――――――――――――――
破線	1	― ― ― ― ― ― ―
点線	2	‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒
一点破線	3	― ‒ ― ‒ ― ‒ ― ‒ ―
二点破線	4	― ‒ ‒ ― ‒ ‒ ― ‒ ‒ ―
透明線	5	横、斜線が表示されません。

＜例＞　太さ3ドットの破線を表示する時には次の様に指定します。



*hmdc　　セル修飾が KTBMES, KTBSLA の時、網掛け又は斜線の色を指定します。

imod　　データ3がON状態になった時に、変更するカーソルタイプ
(挿入/上書き) を示します。

タイプ	値
挿入	KFINS
上書き	KFOVR
変更しない	KFNO

cx　　データ3がON状態になった時のカーソル桁位置を示します。(0～)
入力エリアの最後にカーソルを表示する場合は、 -1 を指定します。
マウスカーソル部にカーソルを表示する場合は、 -2 を指定します。

cy　　データ3がON状態になった時のカーソル行位置を示します。(0～)

*chk　　入力比較文字列 (ワイルドカード文字列) を示します。
入力確定時に入力文字列に対して有効な文字列であるかのチェックを
行ないます。チェックを行なわない場合には NULL を指定します。

*fmt　　NULL を指定します。

*data　　入力データを格納するバッファのポインタを指定します。

注 意　　バッファはデータタイプで指定したデータ型と
同じ型とします。

id　　データ3識別ユーザ定義IDを示します。

*subb　　キー入力開始前、及び終了時に呼び出されるサブルーチンを指定します。
*sube

flg　　キーの入力状態を示します。

キー入力　OFF　･･･　KIOFF
キー入力　ON　･･･　KION

*inf　　ライブラリが部品を管理する為の情報。
初期値は0を指定します。

event　　データ3部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

・ データ3処理 前コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, data3, mask);

    Ktcore      *core;      /*    パネル管理情報                          */
    Ktdata3     *data3;     /*    データ3部品                             */
    int         mask;       /*    データ3選択マウスボタンマスク           */
                    ---->マウス左ボタン ・・・・・・ Button1Mask
                         マウス中ボタン ・・・・・・ Button2Mask
                         マウス右ボタン ・・・・・・ Button3Mask
                         セル移動    ・・・・・・・・・ NULL
```

リターンステータス

```
    0             ・・・  正常終了
    0 より小さい  ・・・  データ3の選択解除
    0 より大きい  ・・・  イベント処理の終了
```

補足:「表示のみ」の場合でも呼ばれます。入力フィールドの切り換えはありません。

・ データ3処理 後コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, data3, type, text);

    Ktcore      *core;      /*    パネル管理情報                          */
    Ktdata3     *data3;     /*    データ3部品                             */
    int         type;       /*    終了タイプ                              */
                    ---->KEYESC     ・・・  エスケープ
                         KEYCR      ・・・  リターン
                         KEYTAB     ・・・  タブ
                         KEYBAT     ・・・  シフト + タブ
                         KEYUP      ・・・  ↑
                         KEYDW      ・・・  ↓
                         KEYRT      ・・・  →
                         KEYLF      ・・・  ←
                         KIMAX      ・・・  確定文字数による確定
                         KEYMUS     ・・・  マウス操作  (環境変数KTISUB設定時のみ)
                         KEYSW      ・・・  関数操作    (環境変数KTISUB設定時のみ)
    char        *text;      /*    入力テキスト                            */
```

リターンステータス

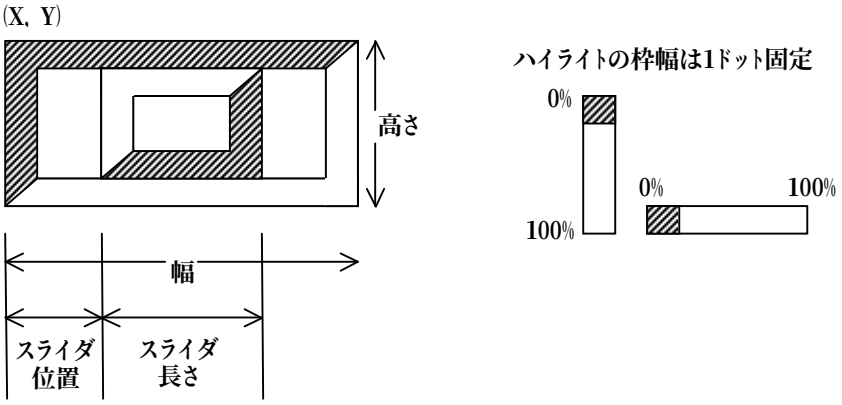
```
    0             ・・・  正常終了
    0 より小さい  ・・・  データ3の再入力
    0 より大きい  ・・・  イベント処理の終了
```

※データ3の表計算機能については新帳票の『**表計算機能**』を御参照下さい。

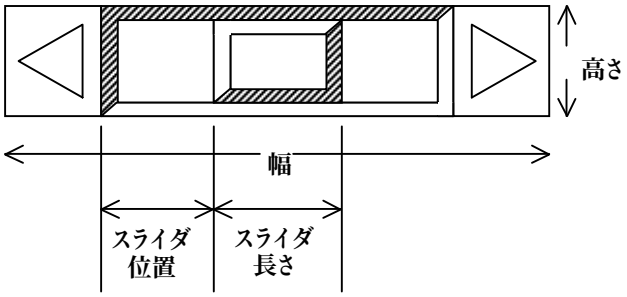
スクロールバー　KOSBAR

機　能

スクロールバーの表示とイベント処理を行ないます。



バーボタン付加の場合



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  スクロールバーモード                */
    int         x;          /*  表示位置  X                         */
    int         y;          /*  表示位置  Y                         */
    int         w;          /*  幅                                  */
    int         h;          /*  高さ                                */
    int         bl;         /*  スライダ長さ                        */
    char        *pc;        /*  バックプレート色                    */
    char        *bc;        /*  スライダ色                          */
    char        *hc1;       /*  ハイライト色1                       */
    char        *hc2;       /*  ハイライト色2                       */
    int         (*sub) ();  /*  コールバックルーチン                */
    int         bp;         /*  スライダ位置                        */
    QID         event;      /*  イベント処理情報                    */
}   Ktsbar;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　スクロールバーの表示形式を以下に示します。

モード	値	説　　明
縦	KBTATE	縦型のスクロールバー
横	KBYOKO	横型のスクロールバー
縦	KBTATE2	縦型のスクロールバー (マウス中ボタンのﾘﾘｰｽ時に コールバックルーチン起動)
横	KBYOKO2	横型のスクロールバー (マウス中ボタンのﾘﾘｰｽ時に コールバックルーチン起動)

スクロールバーの両端にバーボタンを付加した場合は、KBBT と OR をとるとバーボタンが付加出来ます。

(例) 横型スクロールバーにバーボタンを付加する時には次の様に指定します。

KBYOKO |KBBT

x　　スクロールバーの表示位置を示します。
y

w　　スクロールバーの幅、高さを示します。
h

bl　　上位16ビット：スクロールバーの全長の割り合いを示します。
下位16ビット：スライダ部の長さをスクロールバーの幅に対する割り合いを示します。
上位16ビットが、0の時には、全長を100%として扱います

*pc　　スクロールバーのバックプレートとスライダの表示色です。
*bc　　(付録1⑤参照)

*hc1　　スクロールバーのハイライト色を示します。
*hc2

1 ･･･　明部表示色部分の指定。
2 ･･･　暗部表示色部分の指定。
(付録1⑤参照)

*sub　　マウスボタンによるスライダ位置の変更時に、ライブラリから呼び出されるサブルーチンを指定します。
スクロールバーは下記に示すマウスの操作により動作し、これに対応したコールバックルーチンが起動します。

マウス右ボタンクリック。　(スライダがプラス方向に移動)
マウス中ボタンクリック。　(スライダがボタンを押している間マウスに追従して移動)
マウス左ボタンクリック。　(スライダがマイナス方向に移動)

bp　　スライダの位置を示します。

event　　スクロールバー部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

・ スクロールバー処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, bp, sbar);

    Ktcore      *core;      /*    パネル管理情報          */
    int         bp;         /*    スライダ位置            */
    Ktsbar      *sbar;      /*    スクロールバー部品      */
```

リターンステータス

0	・・・	正常終了
0 より小さい	・・・	正常終了
0 より大きい	・・・	イベント処理終了

MEMO

ポインタ KOPOIT

機　能

マウスボタンのｲﾍﾞﾝﾄ処理を行ないます。

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*   ｵﾌﾞｼﾞｪｸﾄﾍｯﾀﾞ                    */
    int         butn;       /*   ﾏｳｽﾎﾞﾀﾝﾏｽｸ                      */
    int         x;          /*   ｲﾍﾞﾝﾄｴﾘｱ位置  X                 */
    int         y;          /*   ｲﾍﾞﾝﾄｴﾘｱ位置  Y                 */
    int         w;          /*   ｲﾍﾞﾝﾄｴﾘｱ  幅                    */
    int         h;          /*   ｲﾍﾞﾝﾄｴﾘｱ  高さ                  */
    XID         pid;        /*   ﾎﾟｲﾝﾀ  ID                       */
    int         (*sub) ();  /*   ｺｰﾙﾊﾞｯｸﾙｰﾁﾝ                    */
    QID         event;      /*   ｲﾍﾞﾝﾄ処理情報                   */
}   Ktpoit;
```

ﾒﾝﾊﾞｰ説明

hed　　ｵﾌﾞｼﾞｪｸﾄのﾍｯﾀﾞ情報を示します。
(2.1部品共通情報参照)

butn　　ﾎﾟｲﾝﾀ指定に使用するﾏｳｽﾎﾞﾀﾝを指定します。
複数指定も可能です。

値	指定ﾎﾞﾀﾝ
Button1Mask	左ﾎﾞﾀﾝ
Button2Mask	中央ﾎﾞﾀﾝ
Button3Mask	右ﾎﾞﾀﾝ
Button123Mask	全ﾎﾞﾀﾝ指定

x
y　　ﾎﾟｲﾝﾀｲﾍﾞﾝﾄ検出ｴﾘｱの座標位置を示します。

w
h　　ﾎﾟｲﾝﾀｲﾍﾞﾝﾄ検出ｴﾘｱの幅、高さを示します。

pid　　ﾎﾟｲﾝﾀｵﾌﾞｼﾞｪｸﾄに対するﾕｰｻﾞ定義IDです。

*sub　　ﾎﾟｲﾝﾀｲﾍﾞﾝﾄに対して呼び出されるｻﾌﾞﾙｰﾁﾝを指定します。

event　　ﾎﾟｲﾝﾀ部品のｲﾍﾞﾝﾄ処理管理情報のQIDを示します。
ﾗｲﾌﾞﾗﾘにて管理する為、初期値は0を指定します。

・ ポインタ処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, x, y, event, poit);

    Ktcore      *core;      /*    パネル管理情報              */
    int         x, y;       /*    マウスカーソル座標          */
    int         event;      /*    イベントタイプ              */
                            → マウスボタン押し下げ ・・・・・・  ButtonPress
                              マウスボタン解放     ・・・・・・  ButtonRelease
    Ktpoit      *poit       /*    ポインタ部品                */
```

リターンステータス

```
0              ・・・  正常終了
0 より小さい   ・・・  次オブジェクトチェック
0 より大きい   ・・・  イベント処理終了
```

ムーブ　KOMOVE

機　能

マウスムーブのイベント処理を行ないます。

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mask;       /*  マウスボタンマスク                  */
    int         x;          /*  イベントエリア位置 X                */
    int         y;          /*  イベントエリア位置 Y                */
    int         w;          /*  イベントエリア 幅                   */
    int         h;          /*  イベントエリア 高さ                 */
    int         (*sub) ();  /*  コールバックルーチン                */
    QID         event;      /*  イベント処理情報                    */
}   Ktmove;
```

メンバー説明

hed　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mask　ムーブで使用するマウスボタンを指定します。
複数指定も可能です。

値	指定ボタン
Button1Mask	左ボタン
Button2Mask	中央ボタン
Button3Mask	右ボタン
Button123Mask	全ボタン指定

NULLを設定した場合は、マウスボタンをクリックしなくても、
エリアにマウスカーソルが進入しただけでイベントを発生する
事が出来ます。

x
y　ムーブイベント検出エリアの座標位置を示します。

w
h　ムーブイベント検出エリアの幅、高さを示します。

*sub　ムーブイベントに対して呼び出されるサブルーチンを指定します。

event　ムーブ部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

・ ムーブ処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, x, y, move);

    Ktcore      *core;      /*   パネル管理情報          */
    int         x, y;       /*   マウスカーソル座標      */
    Ktmove      *move       /*   ムーブ部品              */
```

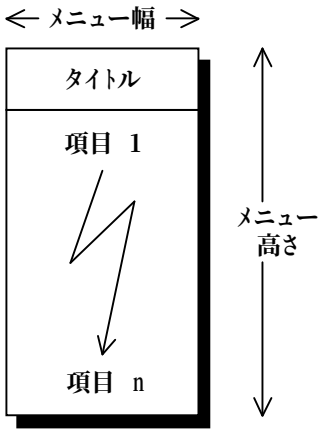
リターンステータス

0	・・・	正常終了
0 より小さい	・・・	次オブジェクトチェック
0 より大きい	・・・	イベント処理終了

プルダウンメニュー　KOMENU

機　能

指定エリア内でマウスをプッシュするとプルダウンメニューを表示し、項目を選択するとイベント処理を呼び出します。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mask;       /*  ボタンマスク                        */
    int         x;          /*  イベントエリア位置 X                */
    int         y;          /*  イベントエリア位置 Y                */
    int         w;          /*  イベントエリア 幅                   */
    int         h;          /*  イベントエリア 高さ                 */
    int         mw;         /*  メニュー 幅                         */
    int         mh;         /*  メニュー 高さ                       */
    int         tfont;      /*  メニュータイトルフォントコード      */
    char        *ttext;     /*  タイトル文字列                      */
    int         ifont;      /*  メニュー項目フォントコード          */
    char        **itext;    /*  項目文字列テーブルアドレス          */
    int         iw;         /*  ビットマップ項目 幅                 */
    int         ih;         /*  ビットマップ項目 高さ               */
    char        *mc;        /*  メニュー背景色                      */
    char        *tc;        /*  タイトル表示色                      */
    char        *ic;        /*  項目表示色                          */
    int         (*sub) ();  /*  コールバックルーチン                */
    int         flg;        /*  管理フラグ                          */
    QID         event;      /*  イベント処理情報                    */
}   Ktmenu;
```

メンバー説明

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mask　　メニューの表示を行なう時に使用するマウスボタンを指定します。
複数指定も可能です。

値	指定ボタン
Button1Mask	左ボタン
Button2Mask	中央ボタン
Button3Mask	右ボタン
Button123Mask	全ボタン指定

x　　メニュー表示イベント検出エリアの座標位置を示します。
y

w　　メニュー表示イベント検出エリアの幅、高さを示します。
h

mw　　表示されるメニューの幅、高さを示します。
mh

tfont　　タイトルの表示に使用するフォントタイプの指定を行ないます。
(付録1④-1参照)

*ttext　　タイトルの表示文字列アドレスを与えます。
タイトルを表示しない場合は、NULL を設定します。

ifont　　メニュー項目の表示に使用するフォントタイプの指定を行ないます。
(付録1④-1, ④-2参照)

**itext　　・ 項目が文字列の場合
表示する文字列テーブルのアドレスを指定します。

・ 項目がビットマップデータの場合
ファイルデータでは、ビットマップファイルPATH配列の
アドレスを指定します。

メモリデータでは、ビットマップデータ変数名配列のアドレス
を指定します。

注 意
どちらの場合もテーブルエンドには、NULL を指定します。

iw　　項目がビットマップデータの場合その大きさを指定します。
ih　　メモリデータの場合は、全項目とも同じ大きさのビットマップデータ
を使用して下さい。
項目のフォントタイプがビットマップデータでない場合、この値は
意味を持ちません。

*mc　　メニューのパネル色を指定します。
(付録1⑤参照)

*tc　　タイトル、項目それぞれの表示色を指定します。
*ic　　(付録1⑤参照)

*sub　　メニュー項目選択時にライブラリから呼び出されるサブルーチンを
指定します。

flg　　ライブラリにて管理する状態フラグ。初期値は0を指定します。

event　　プルダウンメニュー部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

・ プルダウンメニュー処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, ino, x, y, menu);

    Ktcore      *core;      /*   パネル管理情報              */
    int         ino;        /*   選択項目No.(0～)            */
    int         x, y;       /*   マウスポジション            */
    Ktmenu      *menu       /*   プルダウンメニュー部品      */
```

リターンステータス

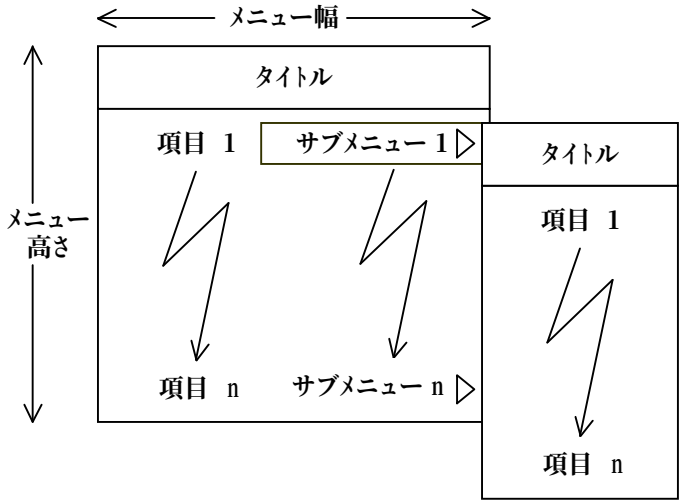
```
    0               ・・・   正常終了
    0 より小さい    ・・・   正常終了
    0 より大きい    ・・・   イベント処理終了
```

MEMO

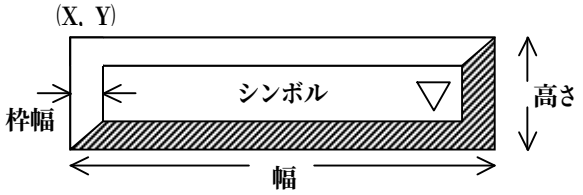
新メニュー　KONMENU

機　能

　指定エリア内（プルダウンメニュー）又は、指定ボタン（メニューボタン）をマウスボタンでクリックするとプルダウンメニューを表示（ポップアップメニューはメニュー状態フラグが1の時に登録、又は、登録後に表示するとプルダウンメニューを表示）し、メニュー表示時とアイテム選択時にイベント処理を行ないます。



メニューボタンモード時



　メニューボタンモード時、この様なボタンが表示され、マウスクリックする事により、上図の様なメニューが表示される。

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                          */
    int         mode;       /*  モード                                      */
    int         mask;       /*  ボタンマスク                                */
    int         x;          /*  イベントエリア位置  X                       */
    int         y;          /*  イベントエリア位置  Y                       */
    int         w;          /*  イベントエリア  幅                          */
    int         h;          /*  イベントエリア  高さ                        */
    int         wh;         /*  メニューボタン枠幅                          */
    int         font;       /*  メニューボタンシンボルフォントコード        */
    char        *sym;       /*  メニューボタンシンボルストリングスアドレス  */
    char        *pc;        /*  メニューボタンボタン色                      */
    char        *sc;        /*  メニューボタンシンボル色                    */
    char        *hc;        /*  メニューボタンハイライト明色                */
    char        *bc;        /*  メニューボタンハイライト暗色                */
    Ktmpane     *pane;      /*  メニュー定義情報アドレス                    */
    XID         id;         /*  新メニューID                                */
    int         (*subb) (); /*  前コールバックルーチン                      */
    int         (*sube) (); /*  後コールバックルーチン                      */
    int         flg;        /*  メニュー状態フラグ                          */
    int         *inf;       /*  部品管理情報                                */
    QID         event;      /*  イベント処理情報                            */
}   Ktnmenu;

typedef struct
{
    int         w;          /*  メニュー幅                                  */
    int         h;          /*  メニュー高さ                                */
    char        *pc;        /*  メニューパネル色                            */
    char        *hc;        /*  ハイライト明色                              */
    char        *bc;        /*  ハイライト暗色                              */
    int         font;       /*  アイテムフォントコード                      */
    char        *titl;      /*  タイトルストリングス                        */
    Ktmitem     *item;      /*  アイテム定義情報アドレス                    */
}   Ktmpane;

typedef struct
{
    int         type;       /*  アイテムタイプ                              */
    char        *sc;        /*  アイテム表示色                              */
    char        *sym;       /*  アイテムストリングス                        */
    char        *pane;      /*  メニュー定義情報アドレス                    */
    XID         id;         /*  アイテムID                                  */
    int         flg;        /*  アイテムフラグ                              */
}   Ktmitem;
```

メンバー説明

新メニュー情報(Ktnmenu)

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　新メニューの表示モードを示します。

値	モード
KMPUL	プルダウンメニュー
KMBTN	メニューボタン(▽マークあり)
KMBTN2	メニューボタン(▽マークなし)
KMPOP	ポップアップメニュー

mask　　メニューの表示を行なう時に使用するマウスボタンを指定します。
複数指定も可能です。

値	指定ボタン
Button1Mask	左ボタン
Button2Mask	中央ボタン
Button3Mask	右ボタン
Button123Mask	全ボタン指定

x　　KMPULモード時　：　メニュー表示イベント検出エリアの座標位置を示します。
y　　KMBTN, KMBTN2モード時：　メニューボタンの表示位置を示します。
KMPOPモード時　：　メニューの表示位置を示します。

w　　KMPULモード時　：　メニュー表示イベント検出エリアの幅、高さを示します。
h　　KMBTN, KMBTN2モード時：　メニューボタンの幅、高さを示します。
KMPOPモード時　：　無効データとなります。

wh　　メニューボタンの枠幅を示します。
KMBTN, KTBTN2モードのみ有効。

font　　メニューボタンのシンボルの表示に使用するフォントタイプの指定を行ないます。
(付録1④-1参照)
KMBTN, KTBTN2モードのみ有効。

*sym　　メニューボタンに表示するシンボルの文字列を指定します。
KMBTN, KTBTN2モードのみ有効。

*pc　　メニューボタンの表示色を示します。
KMBTN, KTBTN2モードのみ有効。

*sc　　メニューボタンのシンボル色を示します。(付録1⑤参照)
KMBTN, KTBTN2モードのみ有効。

*hc　　メニューボタンのハイライト色を示します。(付録1⑤参照)
*bc　　KMBTN, KTBTN2モードのみ有効。
また、KMBTN2モードの際にボタン枠幅(wh)が 0 の時、選択時の
ボタン表示色になります。(hc=パネル色、bc=シンボル色)

*pane　　メニュー定義情報のアドレスを示します。

id　　部品にユーザが定義出来る情報。

*subb　　メニュー表示時にライブラリから呼び出されるサブルーチンを指定します。

*sube　　メニューのアイテム選択時にライブラリから呼び出されるサブルーチンを指定します。

flg　　KMPOPモード時
　　ON　：　登録(TKentry)と同時にメニューを表示する。
　　OFF　：　登録時にメニューは表示しない。
KMBTN, KMBTN2モード時
　　-1　：　表示が見え消し状態になり、操作不能となる。
　　OFF　：　操作可能となる。

*inf　　ライブラリが部品を管理する為の情報。
初期値は0を指定します。

event　　新メニュー部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

メニュー定義情報(Ktmpane)

w
h　　表示するメニューの幅、高さを示します。

*pc　　メニューのパネル色を指定します。
(付録1⑤参照)

*hc
*bc　　メニューのハイライト色を指定します。
(付録1⑤参照)

font　　メニューのアイテムの表示に使用するフォントタイプの指定を行ないます。
(付録1④-1, ④-2参照)

*titl　　タイトルの表示文字列アドレスを与えます。
タイトルが不要な場合は NULL を設定します。

*item　　メニューのアイテム定義情報のアドレスを示します。

アイテム定義情報(Ktmitem)

type　　アイテムのタイプを示します。

値	モード
KMTEXT	テキストアイテム
KMBMAP	ビットマップアイテム
KMSUB	サブメニュー
KMTITL	見出し
KMNEXT	次列
KMNOP	ブランク
KMSEP	区切り
NULL	定義情報の終了

*sc　　アイテムの表示色を示します。
(付録1⑤参照)

*sym　　アイテムとして表示する文字列やビットマップを指定します。
アイテムタイプによって指定する内容が異なります。

KMTITL
KMTEXT　―― アイテムの表示文字列アドレスを与えます。
KMSUB
KMBMAP　―― アイテムの表示ビットマップの構造体。
Ktbitmapの詳細はTkchgbcursor関数説明を参照

*pane　　アイテムタイプが KMSUB の時のみ有効。
サブメニュー定義情報（メニュー定義情報）のアドレスを示します。

id　　アイテム選択時にアプリケーションに通知するID（-1を除く数値）を示します。
アイテムタイプ（type）が次に記すタイプのみ、この設定が有効です。

KMTEXT
KMBMAP
KMSUB

flg　　アイテム選択の有効/無効を示します。

ON　：　有効　（選択可能）
OFF　：　無効　（選択不可能）

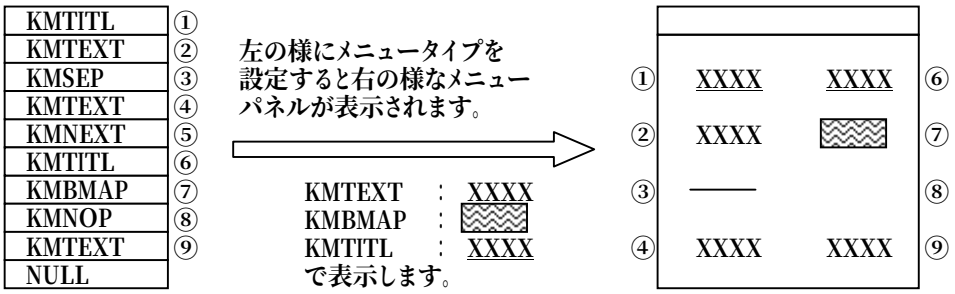
アイテムタイプ（type）が次に記すタイプのみ、この設定が有効です。

KMTEXT
KMBMAP
KMSUB

また、KMTEXT, KMBMP の場合には、次の設定が可能です。

KMCHK　：　チェックマークの表示（機能はONと同じ）

参 考



・ 新メニュー処理 前コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, id, nmenu, pane);

    Ktcore      *core;      /*  パネル管理情報                          */
    int         id;         /*  選択アイテムID                          */
                            /*  サブメニューの場合選択されたアイテムを示し、  */
                            /*  一番最初は -1                           */
    Ktnmenu     *nmenu;     /*  新メニュー部品                          */
    Ktmpane     *pane;      /*  メニュー定義情報                        */
```

リターンステータス

```
0               ・・・  メニュー表示
0 より小さい    ・・・  メニュー非表示、次オブジェクトチェック
0 より大きい    ・・・  イベント処理終了
```

・ 新メニュー処理 後コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, id, nmenu, pane);

    Ktcore      *core;      /*  パネル管理情報                          */
    int         id;         /*  選択アイテムID(非選択時:-1)             */
    Ktnmenu     *nmenu;     /*  新メニュー部品                          */
    Ktmpane     *pane;      /*  メニュー定義情報                        */
```

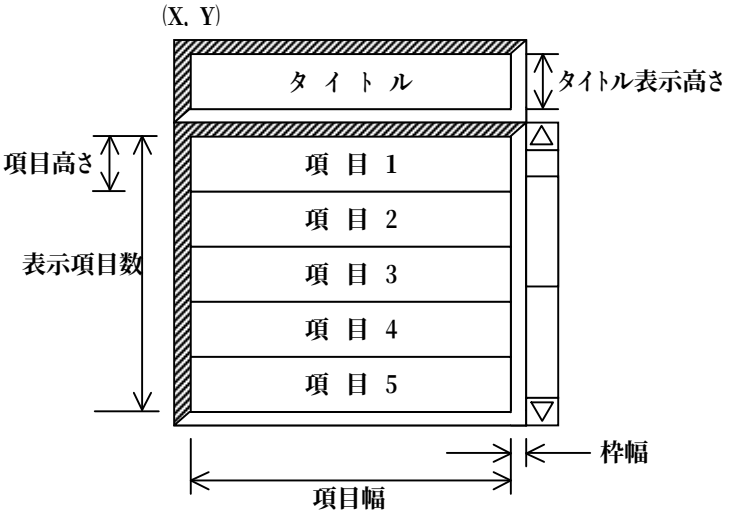
リターンステータス

```
0               ・・・  正常終了
0 より小さい    ・・・  正常終了
0 より大きい    ・・・  イベント処理終了
```

リスト　KOLIST

機　能

リスト表示とイベント処理を行ないます。
　また、全項目数が表示項目数よりも多い場合は、スクロールバーを表示させる事が出来ます。
　このスクロールバーのスライダを移動させると、スライダの動きに合わせて項目の表示部分も移動し、項目全体を見る事が出来ます。
　項目上へのマウスカーソル進入、及びマウスボタン押し下げ/解放によりイベント処理を呼び出します。また、選択出来る項目は個別あるいは複数の指定が可能です。



★ 枠、罫線、タイトル付きの場合

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                  */
    int         mode;       /*  モード                              */
    int         x;          /*  表示位置 X                          */
    int         y;          /*  表示位置 Y                          */
    int         w;          /*  項目幅                              */
    int         h;          /*  項目高さ                            */
    int         hl;         /*  表示項目数                          */
    int         wh;         /*  枠幅                                */
    int         th;         /*  タイトル表示高さ                    */
    int         arow;       /*  表示列数(1固定)                     */
    int         ds;         /*  選択項目表示タイプ                  */
    int         posi;       /*  項目文字表示位置                    */
    int         tfont;      /*  タイトルフォントコード              */
    char        *title;     /*  タイトル文字列                      */
```

```
    int         ifont;      /*  項目フォントコード                  */
    Ktlitem     *item;      /*  アイテム定義情報アドレス            */
    int         stl;        /*  表示開始項目位置(0～)               */
    char        *pc;        /*  スクロールバー色                    */
    char        *hc;        /*  ハイライト明色                      */
    char        *sc;        /*  ハイライト暗色                      */
    char        *kc;        /*  罫線色                              */
    char        *tc;        /*  タイトル文字表示色                  */
    char        *tbc;       /*  タイトル背景色                      */
    XID         id;         /*  リストID                            */
    int         (*sub) ();  /*  コールバックルーチン                */
    int         *inf;       /*  部品管理情報                        */
    QID         event;      /*  イベント処理情報                    */
} Ktlist;

typedef struct
{
    char        *sym;       /*  項目文字列                          */
    char        *fc;        /*  項目文字色                          */
    char        *bc;        /*  項目背景色                          */
    char        *sfc;       /*  選択項目文字色                      */
    char        *sbc;       /*  選択項目背景色                      */
    int         sel;        /*  項目選択フラグ                      */
    XID         id;         /*  項目ID                              */
} Ktlitem;
```

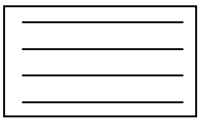
メンバー説明

リスト情報 (Ktlist)

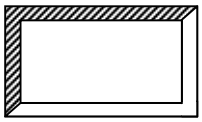
hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

mode　　リスト部品のモードを示します。

モード	値	
罫線付き	KLK	
枠付き	KLW	
スクロール･バー付き	KLS	
複数選択	KLF	未対応



罫線付き



枠付き

スクロールバー付き

補 足

モードは複数のモードをORにより組み合わせる事が出来ます。
(例:罫線、スクロールバー付きの設定は KLK |KLS となります。)

x
y　　リストの表示位置を示します。

w
h　　項目の幅、高さを示します。

注 意

項目高さは、フォントサイズの高さに1を加算した値以上必要です。
フォントサイズの高さに1を加算した値以下に設定した場合は補正されます。

hl　　表示させる項目数を示します。

wh　　ﾘｽﾄの枠幅を示します。

th　　タイトル表示エリアの高さを示します。
タイトルを表示させない場合は0を設定して下さい。

arow　　1固定です。

ds　　ﾘｽﾄ項目選択時に表示する項目形状を設定する。

タイプ	値	
表示なし	KLSN	
反転表示	KLSR	
塗りつぶし	KLSP	未対応
ボタン型	KLSB	未対応
枠型	KLSW	未対応
角丸枠型	KLSK	未対応
端丸枠型	KLSH	未対応

posi　　項目に対する文字の表示位置を示します。

表示位置	値
右詰め	KPR
中央	KPM
左詰め	KPL

tfont　　タイトル文字の表示フォントコードを示します。
(付録1④-1参照)

*title　　表示を行なうタイトル文字のアドレスを示します。

ifont　　項目文字の表示フォントコードを示します。
(付録1④-1参照)

*item　　ﾘｽﾄのアイテム定義情報アドレスを示します。

stl　　表示開始項目を示します。(0～)

*pc　　スクロールバー色を示します。
(付録1⑤参照)

*hc　　枠、スクロールバー等のハイライト表示色を示します。
*sc　　(付録1⑤参照)

*kc　　罫線色を示します。
(付録1⑤参照)

*tc　　タイトルの文字色を示します。
(付録1⑤参照)

*tbc　　タイトルの背景色を示します。
(付録1⑤参照)

id　　ﾘｽﾄ識別ユーザ定義IDを示します。

*sub　　マウスカーソルの項目進入、及びマウスボタン押し下げ/解放時に
呼ばれるサブルーチンを指定します。

*inf　　ライブラリが部品を管理する為の情報。
初期値は0を指定します。

event　　リスト部品のイベント処理管理情報のQIDを示します。
ライブラリにて管理する為、初期値は0を指定します。

アイテム定義情報 (Ktlitem)

*sym　　表示を行なう項目文字のアドレスを示します。
NULL は項目の終了を示します。

*fc　　選択されていない項目文の表示色を示します。
*bc　　(付録1⑤参照)

*sfc　　選択されている項目の表示色を示します。
*sbc　　(付録1⑤参照)

sel　　項目の選択状態を示します。

状態	値
選択されている	ON
選択されていない	OFF

id　　項目選択時にアプリケーションに通知するIDを示します。(0以上)

・ リスト処理コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名 (core, list, line, id, type) ;

    Ktcore      *core;      /*    パネル管理情報          */
    Ktlist      *list;      /*    リスト部品              */
    int         line;       /*    画面上の選択行          */
    int         id;         /*    選択項目ID              */
    int         type;       /*    呼び出し方法通知        */
                  ↓
                ReleaseMove   :  マウスボタンを押さずに、項目にマウス進入
                PressMove     :  マウスボタンを押しながら、項目にマウス進入
                ButtonPress   :  マウスボタン押し下げ
                ButtonRelease :  マウスボタン解放
```

リターンステータス

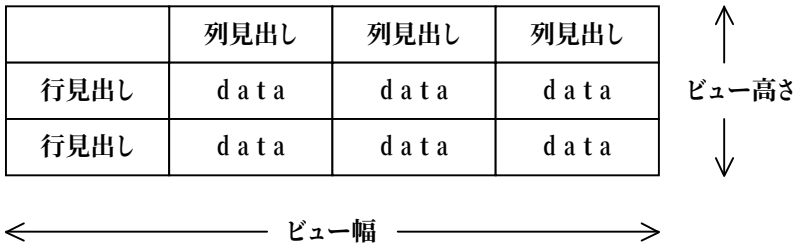
```
    0              ・・・  正常終了
    0 より小さい   ・・・  次オブジェクトチェック
    0 より大きい   ・・・  イベント処理終了
```

帳票　　KOTABLE

機　能

帳票の表示とｲﾍﾞﾝﾄ処理を行ないます。

(X, Y)



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  ｵﾌﾞｼﾞｪｸﾄﾍｯﾀﾞ                    */
    int         x;          /*  表示位置 X                      */
    int         y;          /*  表示位置 Y                      */
    int         mw;         /*  方眼 幅                         */
    int         mh;         /*  方眼 高さ                       */
    int         vw;         /*  ﾋﾞｭｰ 幅                         */
    int         vh;         /*  ﾋﾞｭｰ 高さ                       */
    int         ctc;        /*  列見出し 高さ                   */
    int         ltc;        /*  行見出し 幅                     */
    char        *pc;        /*  表ﾊﾞｯｸ色                        */
    char        *tc;        /*  表影明色                        */
    char        *bc;        /*  表影暗色                        */
    char        *vc;        /*  縦罫線色                        */
    char        *hc;        /*  横罫線色                        */
    int         vp;         /*  縦罫線種                        */
    int         hp;         /*  横罫線種                        */
    Ktcset      *cset;      /*  列見出しﾃｰﾌﾞﾙｱﾄﾞﾚｽ               */
    Ktlset      *lset;      /*  行見出しﾃｰﾌﾞﾙｱﾄﾞﾚｽ               */
    Ktdset      *dset;      /*  表ﾃﾞｰﾀﾃｰﾌﾞﾙｱﾄﾞﾚｽ                */
    int         (*subb) (); /*  前ｺｰﾙﾊﾞｯｸﾙｰﾁﾝ                   */
    int         (*sube) (); /*  後ｺｰﾙﾊﾞｯｸﾙｰﾁﾝ                   */
    int         *inf;       /*  ﾗｲﾌﾞﾗﾘ管理情報                  */
}   Kttable;

typedef struct
{
    int         clum;       /*  列見出し幅                      */
    int         font;       /*  列見出しﾌｫﾝﾄｺｰﾄﾞ                */
    char        *titl;      /*  列見出しﾀｲﾄﾙ文字列               */
    char        *col;       /*  列見出し表示色                  */
    int         posi;       /*  列見出し表示位置                */
}   Ktcset;
```

```
typedef struct
{
    int         line;       /*  行見出し高さ                    */
    int         font;       /*  行見出しフォントコード          */
    char        *titl;      /*  行見出しタイトル文字列          */
    char        *col;       /*  行見出し表示色                  */
    int         posi;       /*  行見出し表示位置                */
}   Ktlset;

typedef struct
{
    int         mode;       /*  データモード                    */
    int         clum;       /*  データ表示文字数                */
    int         font;       /*  データフォントコード            */
    char        *col;       /*  データ表示色                    */
    int         posi;       /*  データ表示位置                  */
    int         type;       /*  データ型                        */
    int         prec;       /*  小数点以下有効桁数              */
    void        *data;      /*  データアドレス                  */
}   Ktdset;
```

メンバー説明

hed	オブジェクトのヘッダ情報を示します。 (2.1部品共通情報参照)
x y	帳票の表示座標位置を示します。
mw mh	方眼の幅、高さを示します。 帳票作成においての最小単位であり、見出し幅、高さやビュー幅、高さ等は、この方眼の個数を指定します。 (単位：ドット)
vw vh	帳票の幅、高さを示します。 このエリア内に納まらない帳票には、スクロールバーが表示されます。 (単位：方眼の個数)
ctc	列見出しの高さを示します。 (単位：方眼の個数)
ltc	行見出しの幅を示します。 (単位：方眼の個数)
*pc	バックの色を示します。 (付録1⑤参照)
*tc *bc	表の明部表示色部分と暗部表示色部分の色を示します。 (付録1⑤参照)
*vc *hc	縦罫線と横罫線の表示色を示します (付録1⑤参照)

vp
hp　線種のモードを以下に示します。

モード	値	説　　明
実線	0	――――――――――――――
破線	1	―― ―― ―― ―― ―― ―― ――
点線	2	－ － － － － － － － － － － － － －
一点破線	3	―― － ―― － ―― － ―― － ――
二点破線	4	―― － － ―― － － ―― － － ――

*cset　列見出しテーブルのアドレスを示します。
テーブルエンドは、列見出し幅に -1 を指定します。

*lset　行見出しテーブルのアドレスを示します。
テーブルエンドは、行見出し高さに -1 を指定します。

*dset　表データテーブルのアドレスを示します。
テーブルエンドは、データモードに -1 を指定します。

*subb　キー入力開始時あるいは、セルをマウスボタンによりクリックした時に呼び出されるサブルーチンを指定します。

*sube　キー入力終了時に呼び出されるサブルーチンを指定します。

*inf　ライブラリで管理する為、初期値は NULL を指定します。

clum
line　列見出しの幅、行見出しの高さを示します。
（単位：方眼の個数）

font　列見出し、行見出し、データのフォントを示すコードです。
（付録1④-1参照）

*titl　列見出し名称、行見出し名称を示します。

*col　列見出し名称、行見出し名称、データの色コードを示します。
（付録1⑤参照）

posi　列見出し名称、行見出し名称、データの表示位置を指定します。

表示位置	値
右詰め	KPR
中央	KPM
左詰め	KPL

mode　データのモードを以下に示します。

説　明	値
表示のみ	KIDDSP
入力あり	KIDIN

type　　表示するデータの型を指定します。

値	表示形式	データ型
KDDEC	10進数	int
KDHEX	16進数	int
KDOCT	8進数	int
KDFLOAT	小数点表示	float
KDEXP	指数表示	float
KDGFORM	上記2つの短い方	float
KDNUM	数字	char
KDANK	アスキー　(日本語不可)	char
KDTXT	テキスト　(日本語可)	char

clum　　表示エリア内への表示文字 (半角) 数を示します。
実際の文字数が表示エリアのサイズを越える場合は、ボタンが表示されスクロールする事が可能となります。
この情報はデータ型がアスキー又はテキストの場合にのみ有効で入力可能な文字数を示します。

prec　　データ型が KDFLOAT, KDEXP, KDGFORM の場合、小数点以下の有効桁数を指定します。
上記3つのデータ型以外は意味を持ちません。

*data　　データを確約する領域のアドレスを示します。
データの内容はデータ型に対応した型の値を確約します。

・ 帳票入力処理 前コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名 (core, table, line, clum, mask) ;

    Ktcore      *core;      /*    パネル管理情報                          */
    Kttable     *table;     /*    帳票部品構造体                          */
    int         line;       /*    選択セル行番号                          */
    int         clum;       /*    選択セル列番号                          */
    int         mask;       /*    セル選択マウスボタンマスク              */
                    ---->   マウス左ボタン ・・・・・・  Button1Mask
                            マウス中ボタン ・・・・・・  Button2Mask
                            マウス右ボタン ・・・・・・  Button3Mask
                            セル移動    ・・・・・・・・・ NULL
```

リターンステータス

```
    0               ・・・  正常終了
    0 より小さい    ・・・  セルの選択解除
    0 より大きい    ・・・  イベント処理終了
```

補足:「表示のみ」の場合でも呼ばれます。入力フィールドの切り換えはありません。

・ 帳票入力処理 後コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, table, line, clum, type, text);

    Ktcore      *core;      /*    パネル管理情報                                    */
    Kttable     *table;     /*    帳票部品構造体                                    */
    int         line;       /*    選択セル行番号                                    */
    int         clum;       /*    選択セル列番号                                    */
    int         type;       /*    終了タイプ                                        */
                    ---->   KEYESC    ・・・  エスケープ
                            KEYCR     ・・・  リターン
                            KEYTAB    ・・・  タブ
                            KEYBAT    ・・・  シフト + タブ
                            KEYUP     ・・・  ↑
                            KEYDW     ・・・  ↓
                            KEYMUS    ・・・  マウス操作   (環境変数KTlSUB設定時のみ)
                            KEYSW     ・・・  関数操作     (環境変数KTlSUB設定時のみ)
    char        *text;      /*    入力テキスト                                      */
```

リターンステータス

```
0               ・・・  正常終了(次セルの入力へ)
                        但し、終了タイプが KEYESC の場合は、セル入力の終了。
0 より小さい    ・・・  セルの再入力
0 より大きい    ・・・  イベント処理終了
```

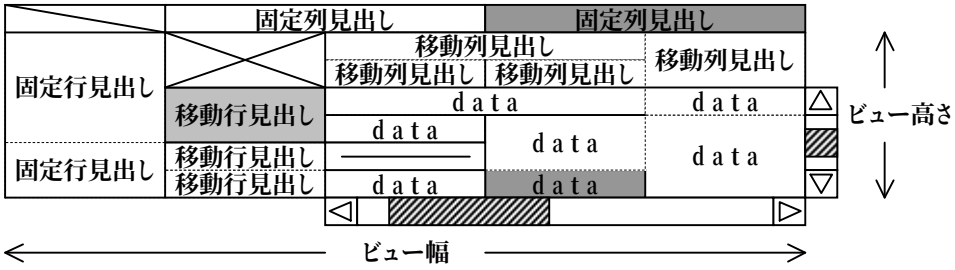
MEMO

新帳票　　KOTABLE2

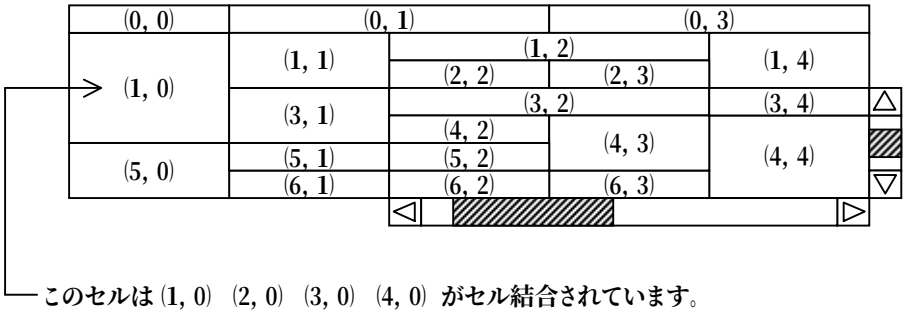
機　能

新帳票の表示とｲﾍﾞﾝﾄ処理を行ないます。

(X. Y)



(参考)　上記の例の新帳票のｾﾙNO. は下記の様になります。



【固定行/列見出し部】　スクロールバーによる移動は行なわれない部分の事です。

【移動行見出し部】　縦スクロールバーによって上下の移動が出来る部分の事です。

【移動列見出し部】　横スクロールバーによって左右の移動が出来る部分の事です。

【data部】　スクロールバーによる上下左右の移動が出来る部分の事です。

固定行/列見出し、移動行/列見出し、data部分は上記の違いのみで、ｲﾍﾞﾝﾄ
の取り込み、キー入力等、同じ機能を持ったセルとして定義されます。

構 造 体

```
typedef struct
{
    Ktobj       hed;        /*  オブジェクトヘッダ                          */
    int         x;          /*  表示位置 X                                  */
    int         y;          /*  表示位置 Y                                  */
    int         wh;         /*  枠幅                                        */
    int         mw;         /*  方眼 幅                                     */
    int         mh;         /*  方眼 高さ                                   */
    int         vw;         /*  ビュー 幅                                   */
    int         vh;         /*  ビュー 高さ                                 */
    int         *cwd;       /*  列セル幅データテーブルアドレス              */
    int         *lhd;       /*  行セル高さデータテーブルアドレス            */
    int         ctl;        /*  固定列見出し 行数                           */
    int         ltc;        /*  固定行見出し 列数                           */
    int         mctl;       /*  移動列見出し 行数                           */
    int         mltc;       /*  移動行見出し 列数                           */
    int         stal;       /*  表示開始セル行                              */
    int         stac;       /*  表示開始セル列                              */
    char        *tc;        /*  表影明色                                    */
    char        *bc;        /*  表影暗色                                    */
    char        *sc;        /*  スクロールバー色                            */
    Ktdtset     *dtest;     /*  セル定義情報アドレス                        */
    char        *dtype;     /*  データ取得タイプ                            */
    int         (*subs) (); /*  スクロールバーコールバックルーチン          */
    int         (*subb) (); /*  前コールバックルーチン                      */
    int         (*sube) (); /*  後コールバックルーチン                      */
    int         *inf;       /*  ライブラリ管理情報                          */
}   Kttable2;
```

```
typedef struct
{
    int         dmode;      /*  データモード                        */
    int         clum;       /*  入力表示桁数                        */
    int         line;       /*  入力表示行数                        */
    int         font;       /*  データフォントコード                */
    char        *col;       /*  データ表示色                        */
    char        *bcol;      /*  データ背景色                        */
    int         posi;       /*  左右データ表示位置                  */
    int         posi2;      /*  上下データ表示位置                  */
    int         type;       /*  データ型                            */
    int         prec;       /*  小数点以下有効桁数                  */
    int         limit;      /*  上下限値チェックモード              */
    double      max;        /*  上限値                              */
    double      min;        /*  下限値                              */
    int         idec;       /*  確定文字数                          */
    int         hmd;        /*  セル修飾                            */
    int         sp;         /*  横、斜線種/太さ                     */
    char        *hmdc;      /*  網掛け/斜線色                       */
    int         imod;       /*  挿入/上書きモード                   */
    int         cx;         /*  カーソル桁位置                      */
    int         cy;         /*  カーソル行位置                      */
    char        *chk;       /*  入力可能文字                        */
    char        *fmd;       /*  表示フォーマット                    */
    int         next;       /*  前、次入力セル指定タイプ            */
    int         ncell;      /*  次入力セル行                        */
    int         ncelc;      /*  次入力セル列                        */
    int         bcell;      /*  前入力セル行                        */
    int         bcelc;      /*  前入力セル列                        */
    int         rjoin;      /*  右セル結合                          */
    int         djoin;      /*  下セル結合                          */
    int         vp;         /*  縦罫線種/太さ                       */
    int         hp;         /*  横罫線種/太さ                       */
    char        *vc;        /*  縦罫線色                            */
    char        *hc;        /*  横罫線色                            */
    int         *data;      /*  データアドレス                      */
    int         cinf;       /*  セル結合管理情報                    */
}   Ktdtset;
```

メンバー説明

新帳票情報 (Kttable2)

hed　　オブジェクトのヘッダ情報を示します。
(2.1部品共通情報参照)

x
y　　新帳票の表示座標位置を示します。

wh　　新帳票の枠幅を示します。

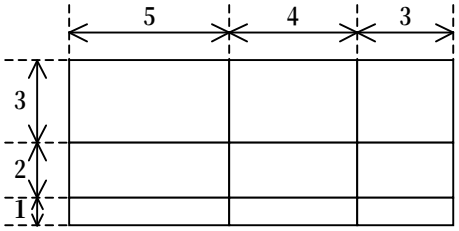
mw
mh　　方眼の幅、高さを示します。
新帳票作成においての最小単位であり、ビュー幅、高さや列セル幅、行セル高さ
データテーブル内の幅、高さ等は、この方眼の個数を指定します。
(単位：ピクセル)

vw
vh　　新帳票の幅、高さを示します。
このエリア内に納まらない新帳票には、スクロールバーが表示されます。
(単位：方眼の個数)

*cwd
*lhd　　見出し、データのセルの基本サイズを示すデータテーブルのアドレス
を示します。(テーブルエンドには、 -1 を指定します。)
この設定によって囲まれた部分がセルになります。
(単位：方眼の個数)

(例)　　列セル幅データテーブル　　行セル高さデータテーブル

上の様にテーブルを設定すると
下の様なセルが展開されます。



ctl　　スクロールバーの動きに伴わない列見出しの行数(セル数)を示します。

ltc　　スクロールバーの動きに伴わない行見出しの列数(セル数)を示します。

mctl　　スクロールバーの動きに伴う列見出しの行数(セル数)を示します。

mltc　　スクロールバーの動きに伴う行見出しの列数(セル数)を示します。

stal
stac　　data部の左上に表示するセルのセル番号を指定します。

*tc
*bc　　表、スクロールバーの明部表示色部分と暗部表示色部分の色を示します。
(付録1⑤参照)

*sc　　スクロールバー色を示します。
(付録1⑤参照)

*dtset　　見出し、表データ定義情報のアドレスを示します。

*dtype　　NULL を指定します。

*subs　　スクロールバーのバー移動時に呼び出されるサブルーチンを指定します。

*subb　　キー入力開始時あるいは、セルをマウスボタンによりクリックした時
に呼び出されるサブルーチンを指定します。

*sube　　キー入力終了時に呼び出されるサブルーチンを指定します。

*inf　　ライブラリで管理する為、初期値は NULL を指定します。

セル定義情報 (Ktdtset)

dmode　　見出し、データのモードを以下に示します。

説　明	値
表示のみ	KIDDSP
入力あり	KIDIN
表計算	KIDEXP

テーブルエンドでは、 -1 を指定します。

clum　　※ 入力表示行数が1の時

表示エリア内への表示文字 (半角) 数を示します。
実際の文字数が表示エリアのサイズを越える場合は、ボタンが
表示されスクロールする事が可能となります。

※ 入力表示行数が2以上の時

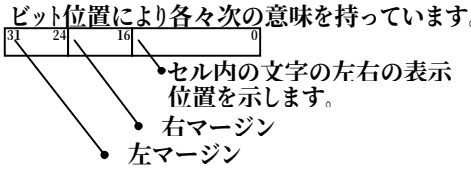
表示エリア内への1行分の表示文字 (半角) 数を示します。
入力表示文字数が表示エリアのサイズを越える場合は、表示エリア
のサイズになります。

line　　表示エリアの行数を示します。
入力表示行数が表示エリアのサイズを越える場合は、表示エリア
のサイズになります。

font　　セル内の文字のフォントを示すコードです。
(付録1④-1参照)

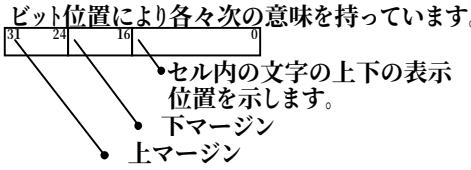
*col
*bcol　　セル内の文字の表示色及び、背景色の色コードを示します。
(付録1⑤参照)

posi　　ビット位置により各々次の意味を持っています。



表示位置	値
右詰め	KPR
中央	KPM
左詰め	KPL

posi2　　ビット位置により各々次の意味を持っています。



表示位置	値
上詰め	KTBUP
中央	KTBMI
下詰め	KTBDW

type　　表示するデータの型を示します。

値	表示形式	データ型
KDDEC	10進数	int
KDHEX	16進数	int
KDOCT	8進数	int
KDUDEC	10進数	unsigned int
KDUHEX	16進数	unsigned int
KDUOCT	8進数	unsigned int
KDSDEC	10進数	short
KDSHEX	16進数	short
KDSOCT	8進数	short
KDUSDEC	10進数	unsigned short
KDUSHEX	16進数	unsigned short
KDUSOCT	8進数	unsigned short
KDFLOAT	小数点表示	float
KDEXP	指数表示	float
KDGFORM	上記2つの短い方	float
KDDOUBL	小数点表示	double
KDDEXP	指数表示	double
KDDGFRM	上記2つの短い方	double
KDNUM	数字	char
KDANK	アスキー　(日本語不可)	char
KDTXT	テキスト　(日本語可)	char
KDPWD	パスワード入力	char

prec　　・ データタイプが KDFLOAT, KDEXP, KDGFORM, KDDOUBL, KDDEXP, KDDGFRM の時、小数点以下の有効桁数を示します。
・ データタイプが KDPWD の時、ANKコード(半角文字)を設定します。
キー入力の際に、この文字で表示されます。ANKコード以外の場合には、デフォルトの '*' で表示します。

limit　　上下限値(数値)チェックの有効/無効を指定します。

値	説　明
KTBLUP	上限値有効
KTBLDWN	下限値有効
NULL	無効

補　足　　上/下限値共に有効にする場合は、KTBLUPとKTBLDWNの OR をとり設定して下さい。(KTBLUP |KTBLDWN)

max　　ここで指定された数値よりも大きい数値は入力出来ません。

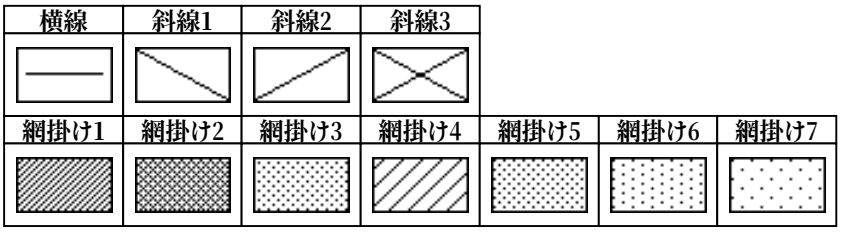
min　　ここで指定された数値よりも小さい数値は入力出来ません。

idec　　入力が指定した文字数に達すると確定となります。
0を設定すると、(データ表示桁数 × データ表示行数) で確定となります。
この設定を無効にしたい場合は、-1を設定して下さい。

hmd　　セルの修飾を指定します。

値	モード
KTBNON	なし
KTBREV	反転
KTBLINE	横線
KTBSLA1	斜線1
KTBSLA2	斜線2
KTBSLA3	斜線3
KTBMES1	網掛け1
KTBMES2	網掛け2
KTBMES3	網掛け3
KTBMES4	網掛け4
KTBMES5	網掛け5
KTBMES6	網掛け6
KTBMES7	網掛け7

各モードによる表示イメージを次に示します。

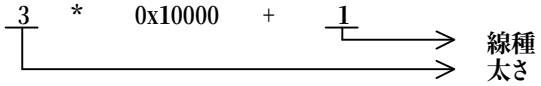


sp　　上位16ビット　・・・　横、斜線の太さを示します。
下位16ビット　・・・　横、斜線種を示します。

線種のモードを以下に示します。

モード	値	説　　明
実線	0	――――――――――――
破線	1	― ― ― ― ― ― ―
点線	2	‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒ ‒
一点破線	3	― ‒ ― ‒ ― ‒ ― ‒ ―
二点破線	4	― ‒ ‒ ― ‒ ‒ ― ‒ ‒ ―
透明線	5	横、斜線が表示されません。

＜例＞　太さ3ドットの破線を表示する時には次の様に指定します。



*hmdc　　網掛け、斜線及び、横線の色を示します。
(付録1⑤参照)

imod　　データ部品がON状態になった時に、変更するカーソルタイプ
(挿入/上書き)を示します。

機能	値
挿入	KFINS
上書き	KFOVR
変更しない	KFNO

cx　　データ部品がON状態になった時のカーソル桁位置を示します。(0～)
入力エリアの最後にカーソルを表示する場合は、 -1 を指定します。
マウスカーソル部にカーソルを表示する場合は、 -2 を指定します。

cy　　データ部品がON状態になった時のカーソル行位置を示します。(0～)

*chk　　入力比較文字列 (ワイルドカード文字列) を示します。
入力確定時に入力文字列に対して有効な文字列であるかのチェックを
行ないます。チェックを行なわない場合には NULL を指定します。

*fmd　　NULL を指定します。

next　　前、次入力セル指定を自セルからの相対行、列数で行なうか、新帳票の左上
のセルからの絶対行、列数で行なうか、又はデフォルトで行なうかを指定します。

値	説　明
KTBORG	絶対　行、列
KTBPRV	相対　行、列
KTBDEF	デフォルト

注　意　　デフォルトとは、リターンキー又は、タブキーが入力
された時に右隣り、シフト + タブキーが入力された時
に左隣りのセルに順次移動する事を示します。

ncell
ncelc　　入力完了でリターンキーまたは、タブキーが入力された時に入力状態に
するセルの行、列数を指定します。

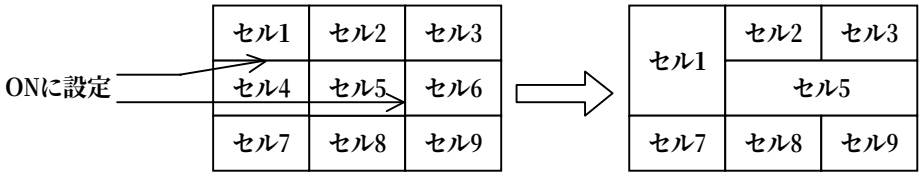
注　意　　存在しないセルを指定した場合は、前、次入力指定タイプ
にデフォルトを設定した時と同じ動きとなります。

bcell
bcelc　　入力完了でシフト + タブキーが入力された時に入力状態にするセルの
行、列数を指定します。

注　意　　存在しないセルを指定した場合は、前、次入力指定タイプ
にデフォルトを設定した時と同じ動きとなります。

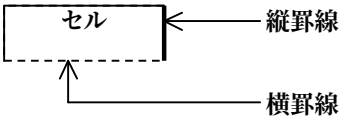
rjoin　　OFF　:　セル結合はしません。
djoin　　ON　:　右又は、下にあるセルと結合します。

セルの結合とは、自セルと対象のセルで1つのセルにする事です。



注　意　　固定見出し部、移動見出し部、データ部にまたがって
のセル結合は出来ません。

vp
hp



設定方法は、横、斜線種/太さの項目を参照して下さい。

*vc　　縦罫線と横罫線の表示色を示します。
*hc　　(付録1⑤参照)

*data　　データを確約する領域のアドレスを示します。
　　データの内容はデータ型に対応した型の値を確約します。

cinf　　ライブラリで管理する為、初期値は NULL を指定します。

・ 新帳票スクロール処理　スクロールバーコールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, table2, line, clum);

    Ktcore      *core;      /*    パネル管理情報                    */
    Kttable2    *table2;    /*    新帳票部品                        */
    int         line;       /*    表示開始セル行番号                */
    int         clum;       /*    表示開始セル列番号                */
```

リターンステータス

```
    0               ・・・   正常終了
    0 より小さい    ・・・   正常終了
    0 より大きい    ・・・   イベント処理終了
```

・ 新帳票入力処理 前コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, table2, line, clum, mask);

    Ktcore      *core;      /*    パネル管理情報                    */
    Kttable2    *table2;    /*    新帳票部品構造体                  */
    int         line;       /*    選択セル行番号                    */
    int         clum;       /*    選択セル列番号                    */
    int         mask;       /*    セル選択マウスボタンマスク        */
                    ---> マウス左ボタン ・・・・・・ Button1Mask
                         マウス中ボタン ・・・・・・ Button2Mask
                         マウス右ボタン ・・・・・・ Button3Mask
                         セル移動   ・・・・・・・・ NULL
```

リターンステータス

```
    0               ・・・   正常終了
    0 より小さい    ・・・   セルの選択解除
    0 より大きい    ・・・   イベント処理終了
```

補足:「表示のみ」の場合でも呼ばれます。入力フィールドの切り換えはありません。

・ 新帳票入力処理 後コールバックルーチン

コーリングシーケンス

```
int     サブルーチン名(core, table2, line, clum, type, text);

    Ktcore      *core;      /*    パネル管理情報                                    */
    Kttable2    *table2;    /*    新帳票部品構造体                                  */
    int         line;       /*    選択セル行番号                                    */
    int         clum;       /*    選択セル列番号                                    */
    int         type;       /*    終了タイプ                                        */
                            ---->  KEYESC    ・・・  エスケープ
                                   KEYCR     ・・・  リターン
                                   KEYTAB    ・・・  タブ
                                   KEYBAT    ・・・  シフト + タブ
                                   KEYUP     ・・・  ↑
                                   KEYDW     ・・・  ↓
                                   KEYRT     ・・・  →
                                   KEYLF     ・・・  ←
                                   KIMAX     ・・・  確定文字数による確定
                                   KEYMUS    ・・・  マウス操作  (環境変数KTISUB設定時のみ)
                                   KEYSW     ・・・  関数操作    (環境変数KTISUB設定時のみ)
    char        *text;      /*    入力テキスト                                      */
```

リターンステータス

```
    0               ・・・  正常終了(次セルの入力)
                            但し、終了タイプが KEYESC の場合は、セル入力の終了
    0 より小さい    ・・・  セルの再入力
    0 より大きい    ・・・  イベント処理終了
```

『表計算機能』

新帳票、DATA3部品において、計算式の設定が可能となりました。
この表計算機能は、帳票に限らず他の部品の値も参照する事が出来ます。
演算結果は自部品のデータ領域に保持され、アプリケーションから参照可能です。
演算の対象となる部品は、数値タイプ（帳票、新帳票、IDATA、DATA3）の部品です。
各部品の表計算機能の設定は次の通りです。

【新帳票】　table2_cell[0][0].dmode = KIDEXP;
　　table2_cell[0][0].chk = "式";

【DATA3】　data3.mode ┠ KIDEXP;
　　data3.chk = "式";

『式』の記述は次の通りです。

【演算子】

+	・・・・・・	加算
-	・・・・・・	減算
*	・・・・・・	乗算
/	・・・・・・	除算
**	・・・・・・	べき乗

【数値】
　　数字　及び　指数表記　（例："1.23E-10"）

【ラベル】
　　データ入力部品　（例："label"）
　　帳票部品　（例："table[行番号][列番号]"）

【カッコ】
　　"（"　及び　"）"　ネストは32段まで有効

【その他】
　　式中のスペースは、サプレスされて評価されます。
　　参照を行なう部品は画面に登録されている事。

【エラー】
　　式にエラーが発見された場合には、"??" が表示されます。
　　また、標準出力に エラーメッセージ と エラーNo. を表示します。
　　以下に、エラーNo. 詳細を記します。

1・・・　内部スタックオーバー
2・・・　数値表記エラー　（小数点重複）
3・・・　数値表記エラー　（指数表記）
4・・・　数値表記エラー　（数字なし）
5・・・　シンボル表記エラー　（インデックス表記）
6・・・　シンボル表記エラー　（データ型）
7・・・　シンボル表記エラー　（部品種別）
8・・・　シンボル表記エラー　（未登録部品）
9・・・　式未結
10・・・　ネストエラー（32まで）
11・・・　ラベル数値なし
12・・・　演算子エラー
13・・・　式なし
14・・・　自部品参照

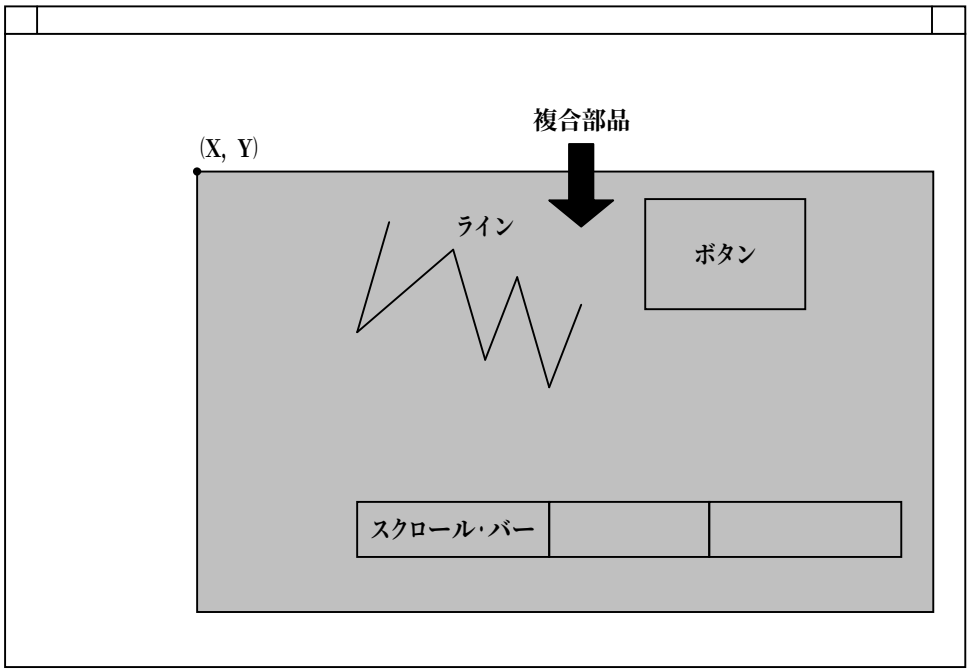
ラベル参照を行ないますので、『ラベル登録関数（TKlabelent）』により予め、
ラベル情報を登録しておく必要があります。

MEMO

複合部品　　KOCOMB

機　能

複数の部品を1つの単体部品として取り扱います。
登録等の指定は、KSINGL で行ないます。



構 造 体

```
typedef struct
{
    Ktobj       hed;        /*    オブジェクトヘッダ                    */
    int         x;          /*    表示位置 X                            */
    int         y;          /*    表示位置 Y                            */
    Ktobj       **obj;      /*    オブジェクトtreeアドレス              */
}   Ktcomb;
```

メンバー説明

hed	オブジェクトのヘッダ情報を示します。 (2.1部品共通情報参照)
x y	複合部品の表示座標位置を示します。 オブジェクト部品群の表示位置 X, Y は、複合部品の表示位置 X, Y を原点とした相対座標を指定します。
**obj	複合部品として登録したいオブジェクトのアドレステーブルを指定します。 テーブルエンドは NULL を指定します。

3 関数情報

X-Mate開発ライブラリの提供関数として以下のものがあります。

- TKinit　　ライブラリ初期設定
- TKexit　　ライブラリ終了
- TKopen　　パネルオープン
- TKiopen　　アイコン状態パネルオープン
- TKuopen　　アンマップ状態パネルオープン
- TKclose　　パネルクローズ
- TKnotify　　パネルノーティファイ設定
- TKiconning　　アイコン状態制御
- TKmapping　　パネル状態制御
- TKentry　　部品の登録
- TKdelete　　部品の削除
- TKdisplay　　部品の表示
- TKerase　　部品の消去
- TKevent　　イベント待ち
- TKevdispatch　　イベントディスパッチ
- TKevmask　　イベントマスク機能
- TKsend　　メッセージ送信
- TKkeysw　　入力部品状態切り換え
- TKkeyoff　　パネル内入力受付状態解除
- TKimode　　入力モード切り換え
- TKpastlock　　コピー&ペースト機能制御
- TKgetbuff　　コピー文字情報取得
- TKchgcell　　帳票セルデータ再表示
- TKwarp　　マウスカーソルの移動
- TKchgcursor　　マウスカーソルの変更
- TKchgbcursor　　ビットマップカーソル
- TKmenucursor　　メニュー部品のカーソル指定
- TKkill　　子ウィンドウ削除
- TKchildkill　　全子ウィンドウ削除
- TKchgpanel　　パネル設定変更
- TKaddfont　　フォント種別追加
- TKclear　　パネルクリア
- TKcanvcore　　キャンバスcore抽出
- CWexec　　子ウィンドウ起動
- CWgetarg　　ウィンドウ起動情報取得
- TKchgcursor2　　マウスカーソル変更2
- TKwildchk　　ワイルドカード関数
- TKlabelent　　ラベル登録関数
- TKlabeldel　　ラベル削除関数
- TKkeyposi　　入力位置取得関数
- TKfepmode　　入力状態取得関数
- Tksbarinc　　スクロールバー部品スクロール量変更関数
- Tklistinc　　リスト部品スクロール量変更関数
- Tkrastinc　　ラスター部品スクロール量変更関数
- Tkcanvinc　　キャンバス部品スクロール量変更関数
- Tktablinc　　帳票部品スクロール量変更関数
- TKtbl2inc　　新帳票部品スクロール量変更関数
- Tkntextinc　　新テキスト部品スクロール量変更関数

ライブラリ初期設定　TKinit

機　能

ライブラリの初期設定を行ないます。

1プロセスに一回行ないます。

コーリングシーケンス

```
Ktroot      *TKinit  ( );        /*   ライブラリルート情報              */
```

戻り値

*TKinit	初期設定されたライブラリのルート情報を示します。 値 0 は、イニシャル出来なかった事を示します。

ライブラリ終了　TKexit

機　能

ライブラリの終了処理を行ないます。

プロセス終了時に行ないます。

コーリングシーケンス

```
TKexit　(root);
    Ktroot　　*root;　　/*　ライブラリルート情報　　*/
```

パラメータ説明

*root　　TKinit（ライブラリ初期設定）で返された情報を渡します。

パネルオープン　TKopen

機　能

パネル（ウィンドウ）をオープンします。
任意のパネルとの親子関係を持たす事が可能です。親パネルが閉じられた際には、自動的に閉じる仕組みを持ちます。

パネルタイプがシステムダイアログ（KSDLG）の際には、マウスのボタンを押されたままになっていると、オープン出来ません。

コーリングシーケンス

```
Ktcore      *TKopen     (root, par, panel) ;

            Ktroot      *root;      /*  ライブラリルート情報        */
            Window      par;        /*  親パネルウィンドウID        */
            Ktpanel     *panel;     /*  パネル定義情報              */
```

戻り値

パネル管理情報　ライブラリが管理するパネルの情報です。
パネルに対して操作する際に使用します。
値 0 は、オープン出来なかった事を示します。

パラメータ説明

*root　TKinit 関数で返された情報を渡します。

par
- 親パネルのウィンドウIDを指定します。自パネルのみ又は親子関係を持たせない場合には、NULL を指定します。
- パネルタイプがサブパネル（KSUBP）の時は親パネルのウィンドウ内にオープンされます。
- パネルタイプが KSDLG, KADLG の時は、無意味となります。

*panel　パネルの情報（大きさ、名称、色、タイトル等）を定義します。

パネル構造体

```
typedef struct
{
    int         type;       /*  パネルタイプ                        */
    int         wx;         /*  パネル表示位置  X                   */
    int         wy;         /*  パネル表示位置  Y                   */
    int         ww;         /*  パネル  幅                          */
    int         wh;         /*  パネル  高さ                        */
    int         bw;         /*  パネル  枠幅                        */
    int         fle;        /*  タイトルバーフラグ    (クローズ)    */
    int         flt;        /*  タイトルバーフラグ    (タイトル)    */
    int         fli;        /*  タイトルバーフラグ    (アイコン)    */
    char        *apl;       /*  パネル名称アドレス                  */
    char        *title;     /*  タイトル文字列                      */
    char        *isym;      /*  アイコン用bitmapファイル名          */
    char        *bg;        /*  パネル背景色                        */
    char        *bd;        /*  パネル枠色                          */
    char        *tl;        /*  パネルタイトルバー色                */
    int         cid;        /*  カーソルID                          */
}   Ktpanel;
```

type　　パネルタイプを以下に示します。

値	名 称	説 明
KPROC	パネル	通常のパネル
KSUBP	サブパネル	指定パネル内にウィンドウを開く
KSDLG	システムダイアログ	自パネル以外の全画面操作禁止
KADLG	アプリケーションダイアログ	プログラム内のみで自パネル以外操作禁止
KOTHER	ウィンドウマネージャ依存パネル	ウィンドウマネージャ特有の表示。KPROC同機能
KPROC_P KSUBP_P KSDLG_P KADLG_P	バックグラウンド描画パネル	上記各パネル機能でバックグラウンドでの 描画を行なった後に画面更新を行なう 再描画時のチラツキに有効。但し、描画 速度の低下が発生。
KPXMP	pixmapパネル	パネルのイメージをメモリに展開。表示なし

(補足)　KPXMPタイプは、Xlibプログラミングの知識を熟知している必要があります。
通常のアプリケーションでは必要ありません。

wx
wy　　画面上のパネル表示位置 (ピクセル座標) を示します。

ww
wh　　パネルの幅、高さを示します。
実際の幅、高さには枠幅が加えられます。

bw　　パネルの枠幅を指定します。

fle
flt
fli　　クローズボタン (fle) 、タイトル表示部分 (flt) 、アイコンボタン (fli)
の有無を指定します。

ON　・・・　あり
OFF　・・・　なし
-1　・・・　パネル変形禁止 (fltのみ有効)

パネルタイプが KSUBP の時は、このフラグに関係なくボタン類は
表示されません。また、 KSDLG, KADLG の時には、アイコンボタンは
表示されません。

*apl　　パネルの名称をANK文字にて設定します。
パネルタイプが KOTHER の際には、次の機能を持ちます。
Motif/OpenWindows 等のウィンドウマネージャの場合
ウィンドウマネージャのフレームタイトルに表示されます。
また、アイコン化を行なった際にも表示されます。
uwn ウィンドウマネージャの場合
アイコン用のビットマップファイル名称が設定されない場合に
アイコン名称として使用されます。

*title　　タイトルバーに表示する文字列を設定します。
タイトルを表示しない場合は、 NULL を指定します。

*isym　　アイコン用のビットマップファイルのパス名称を指定します。
アイコンを使用しない場合は、 NULL を指定します。

*bg
*bd
*tl　　パネルの背景 (bg) 、枠 (bd) 、タイトルバー (tl) の表示色を指定します。
(付録1⑤参照)

cid　　このパネル上でのカーソルの種類を、Xが準備しているcoursorfont.h
内で定義されたIDを用いて指定します。

アイコン状態パネルオープン　　TKiopen

機　能

パネル(ウィンドウ)をアイコン状態でオープンします。

コーリングシーケンス

```
Ktcore      *TKiopen    (root, par, panel, ix, iy) ;

    Ktroot      *root;      /*  ライブラリルート情報        */
    Window      par;        /*  親パネルウィンドウID        */
    Ktpanel     *panel;     /*  パネル定義情報              */
    int         ix;         /*  アイコンポジション X        */
    int         iy;         /*  アイコンポジション Y        */
```

戻り値

パネル管理情報　ライブラリが管理するパネルの情報です。
パネルに対して操作する際に使用します。
値 0 は、オープン出来なかった事を示します。

パラメータ説明

*root　　TKinit 関数で返された情報を渡します。

par　　・ 親パネルのウィンドウIDを指定します。自パネルのみ又は親子関係を持たせない場合には、 NULL を指定します。
・ パネルタイプがサブパネル(KSUBP)の時は親パネルのウィンドウ内にオープンされます。
・ パネルタイプが KSDLG, KADLG の時は、無意味となります。

*panel　　パネルの情報(大きさ、名称、色、タイトル等)を定義します。

ix
iy　　画面上のアイコン表示位置(ピクセル座標)を示します。

アンマップ状態パネルオープン　　TKuopen

機　能

パネルをアンマップ状態でオープンします。
ダイアログの場合、マップする事によりダイアログ機能が働きます。

コーリングシーケンス

```
Ktcore      *TKuopen  (root, par, panel) ;

        Ktroot      *root;      /*   ライブラリルート情報          */
        Window      par;        /*   親パネルウィンドウID          */
        Ktpanel     *panel;     /*   パネル定義情報                */
```

戻り値

パネル管理情報　ライブラリが管理するパネルの情報です。
パネルに対して操作する際に使用します。
値 0 は、オープン出来なかった事を示します。

パラメータ説明

*root　　TKinit 関数で返された情報を渡します。

par　　・ 親パネルのウィンドウIDを指定します。自パネルのみまたは親子関係を持たせない場合には、 NULL を指定します。
・ パネルタイプがサブパネル (KSUBP) の時は親パネルのウィンドウ内にオープンされます。
・ パネルタイプが KSDLG, KADLG の時は、無意味となります。

*panel　　パネルの情報 (大きさ、名称、色、タイトル等) を定義します。

パネルクローズ　TKclose

機　能

オープンしているパネルをクローズします。

コーリングシーケンス

```
TKclose (core);
    Ktcore      *core;          /*  パネル管理情報              */
```

パラメータ説明

*core　TKopen 関数で返された情報を渡します。

パネルノーティファイ設定　　TKnotify

機　能

パネルに発生する各種イベントに対応するアプリケーションの処理を登録します。

コーリングシーケンス

```
*TKnotify    (core, type, sub);

    Ktcore      *core;          /*   パネル管理情報              */
    int         type;           /*   通知タイプ                  */
    int         (*sub)();       /*   コールバックサブルーチン    */
```

パラメータ説明

*core　　対象となるパネルのパネル管理情報を渡します。

type　　通知タイプを以下に示します。

値	説　　明
KEXIT	終了イベント(クローズボタン選択時)
KMESG	メッセージ受信イベント
KEVNT	パネル操作イベント
KKEY	キー入力イベント
KXEV	Xイベント(パネルに発生した全てのイベント)
KVABL	パネルビュアブルイベント(初回オープン通知)

*sub　　上記の通知タイプのイベント発生により呼び出されるサブルーチンを示します。
イベント処理を解除する際には、NULL を指定します。

※ コーリングシーケンスは、次頁以降を参照して下さい。

KEXIT:終了コールバックサブルーチン

機　能

パネル左上にあるクローズボタンが押された時に呼び出されます。

コーリングシーケンス

```
int             sub (core);

        Ktcore      *core;          /*    パネル管理情報                */
```

戻り値

0 より大きい　・・・　イベント処理を終了します。
0　　・・・　パネルをクローズする。
0 より小さい　・・・　パネルをクローズしない。

パラメータ説明

*core　　クローズボタンが押されたパネルのパネル管理情報を示します。

KMESG:メッセージ受信コールバックサブルーチン

機　能

ウィンドウ間（プロセス間）通信でのメッセージ受信時に呼び出されます。

コーリングシーケンス

```
int         sub（core, id, wid, data）;

    Ktcore      *core;      /*  パネル管理情報          */
    int         id;         /*  受信メッセージID        */
    Window      wid;        /*  ウィンドウID            */
    char        *data;      /*  通信データ              */
```

戻り値

0 より大きい　・・・　イベント処理を終了します。
0　・・・　正常終了。（イベント処理を継続）
0 より小さい　・・・　正常終了。（イベント処理を継続）

パラメータ説明

*core　メッセージ受信が発生したパネルのパネル管理情報を示します。

id　受信したメッセージの種別を示します。
種別はアプリケーションで定義して下さい。（TKsend関数使用時）
指定出来る値は、0以上です。

wid　受信したメッセージの送信元パネルのウィンドウIDを示します。

*data　送られてきた通信データのアドレスを示します。

KEVNT：パネル操作イベントコールバックサブルーチン

機　能

パネルに対して移動/変形等のイベントが発生した際に呼び出されます。

コーリングシーケンス

```
int         sub (core, type, info) ;

    Ktcore      *core;      /*  パネル管理情報          */
    int         type;       /*  イベントタイプ          */
    Ktconfig    *info;      /*  ウィンドウ情報          */
```

戻り値

0 より大きい	・・・	イベント処理を終了します。
0	・・・	正常終了。(イベント処理を継続)
0 より小さい	・・・	正常終了。(イベント処理を継続)

パラメータ説明

*core　　イベントが発生したパネルのパネル管理情報を示します。

type　　イベントには下表の種類があります。

値	説　　明
CONFIG	パネルの大きさ、位置、重なりの変化があった時
EXPOSE	パネルが表示された時。又は、隠されていた部分があらわれた時。

*info　　(イベントタイプが CONFIG時のみ有効)

構 造 体

```
typedef struct
{
    int         wx;         /*  パネル表示位置  X       */
    int         wy;         /*  パネル表示位置  Y       */
    int         ww;         /*  パネル  幅              */
    int         wh;         /*  パネル  高さ            */
    Window      above;      /*  ウィンドウID            */
}   Ktconfig;
```

メンバー説明

wx
wy　　画面上のパネルの表示位置 (ピクセル座標) を示します。

ww
wh　　パネルの幅、高さを示します。

above　　そのパネルが一番下にあるならば0、パネルが一番下にない場合は、
そのパネルのすぐ下にあるウィンドウのウィンドウIDがセットされています。

KKEY：キー入力コールバックサブルーチン

機　能

　パネルに発生したキー入力毎に呼び出され、部品に渡される前にアプリケーションで入力文字チェックを行う事が出来ます。
　このイベント処理はキー入力部品が入力状態（かつマウスがパネル上にある）あるいは、パネルに対してキーフォーカスが取得されている場合にのみ呼ばれます。

コーリングシーケンス

```
int         sub (core, sym, stat);

    Ktcore      *core;      /*  パネル管理情報          */
    KeySym      sym;        /*  キーシンボル            */
    int         stat;       /*  修飾キー状態            */
```

戻り値

0 より大きい　・・・　イベント処理を終了します。
0　・・・　そのキーの入力を許可します。
0 より小さい　・・・　そのキーの入力を無視します。

パラメータ説明

*core　キー入力イベントが発生したパネルのパネル管理情報を示します。

sym　入力されたキーのコードを示します。

マシンによってキーのコードが違いますので、
　% xmodmap -pk [Return]
によりリスト出力される Keysym コードを参照して下さい。

stat　修飾キーのマスクパターンを以下に示します。

値	説　明
ShiftMask	Shiftキー
LockMask	Lockキー
ControlMask	Controlキー
ModlMask	Altキー

KXEV：Xイベントコールバックサブルーチン

機　能

パネルに発生した全てのｲﾍﾞﾝﾄで呼び出されます。

コーリングシーケンス

```
int         sub (core, event) ;

    Ktcore      *core;          /*   パネル管理情報              */
    XEvent      *event;         /*   Xｲﾍﾞﾝﾄ構造体               */
```

戻り値

0 より大きい　・・・　ｲﾍﾞﾝﾄ処理を終了します。
0　　・・・　ライブラﾘでｲﾍﾞﾝﾄを処理します。
0 より小さい　・・・　ライブラﾘでｲﾍﾞﾝﾄを処理しません。

パラメータ説明

*core　　ｲﾍﾞﾝﾄが発生したパネルのパネル管理情報を示します。

*event　　詳しくは、Xのマニュアルを参照して下さい。

注　意

この機能は、X Window System を詳しく理解している方のみご利用下さい。

KVABL：パネルビュアブルコールバックサブルーチン

機　能

パネルをオープンし、表示可能になった時に1回だけ呼び出されます。
イニシャル等を行なう場合に使用します。
TKiopen関数、TKuopen関数等でオープンした際には、初めてパネルが可視化
された際に呼ばれます。

コーリングシーケンス

```
int             sub (core);

        Ktcore      *core;          /*    パネル管理情報                  */
```

戻り値

0 より大きい　・・・　イベント処理を終了します。
0　　・・・　イベント処理を継続。
0 より小さい　・・・　イベント処理を継続。

パラメータ説明

*core　　表示可能となったパネルのパネル管理情報を示します。

アイコン状態制御　　TKiconning

機　能

アイコン状態/パネル状態の切り換えを行ないます。

コーリングシーケンス

```
void   TKiconning  (core, mode) ;

       Ktcore      *core;      /*   パネル管理情報          */
       int         mode;       /*   制御モード              */
```

パラメータ説明

*core	TKopen 関数で返された情報を渡します。	
mode	ON	アイコン化 パネルをアイコンに切り換えます。
	OFF	非アイコン化 アイコンをパネルに切り換えます。

パネル状態制御　TKmapping

機　能

パネルの可視化（マップ）/不可視化（アンマップ）の切り換えを行ないます。
ダイアログパネルの場合は、ダイアログ機能の設定/解除となります。

コーリングシーケンス

```
int    TKmapping  (core, mode) ;

       Ktcore     *core;      /*   パネル管理情報          */
       int        mode;       /*   制御モード              */
```

パラメータ説明

*core	TKopen 関数で返された情報を渡します。	
mode	ON	パネルを可視化（マップ）する。
	OFF	パネルを不可視化（アンマップ）する。

戻り値

0　… パネルのマッピング処理が正常。
-1　… パネル管理情報が不正であるか、システムダイアログパネルを可視化する際に既に他のプロセスがシステムダイアログパネルを開いている状態を示します。

部品の登録　TKentry

機　能

指定されたパネルに部品を登録し、表示を行ないます。
部品を登録する事により、再描画等が必要になった際に自動的に行ないます。
既に登録済みの部品を指定した際には、無意味となり何も行ないません。

コーリングシーケンス

```
void   TKentry     (core, mode, obj) ;

       Ktcore      *core;      /*   パネル管理情報          */
       int         mode;       /*   登録モード              */
       Ktobj       *obj;       /*   登録部品データ          */
```

パラメータ説明

*core		部品登録を行ないたいパネルのパネル管理情報を渡します。
mode	KSINGL	単体登録モード 1部品のみの登録を行ないます。
	KARRAY	複数登録モード 部品のアドレステーブルのアドレスをパラメータ として指定し、複数の部品の登録を行ないます。
*obj		単体登録モード時は、登録部品データのアドレスを指定します。 複数登録モード時は、部品のアドレステーブルのアドレスを指定し、 NULL までが有効データとなります。

注 意

パネルオープン直後のTKentryした部品は、ライブラリ内部管理のみ行ない、表示は行ないません。これは X サーバーとの関係上、TKopen関数から戻ってもパネルが存在している訳ではないので、部品を表示する事が出来ません。

パネルをオープンし部品登録後、直ちにイベントループに戻る様にして下さい。
もし、連続して長い処理を行ないたい場合には、下記のいずれかの手順で行なって下さい。

(1)　自分自身にメッセージ送信 (TKsend) を行ない、メッセージ受信処理で行なう。
(2)　パネルビュアブル処理で行なう。

部品の削除　TKdelete

機　能

指定された部品を管理情報から削除し、パネルから消去します。
登録されていない部品を指定した際には、無意味となり何も行ないません。

コーリングシーケンス

```
void   TKdelete    (core, mode, obj) ;

       Ktcore      *core;      /*   パネル管理情報          */
       int         mode;       /*   削除モード              */
       Ktobj       *obj;       /*   削除部品データ          */
```

パラメータ説明

*core　TKopen 関数で返された情報を渡します。

mode　KSINGL　単体削除モード
1部品のみの削除を行ないます。

KARRAY　複数削除モード
部品のアドレステーブルのアドレスをパラメータ
として指定し、複数の部品の削除を行ないます。

KALL　全削除モード
パネルに登録されている全ての部品の削除を
行ないます。

*obj　単体削除モード時は、削除部品データのアドレスを指定します。

複数削除モード時は、部品のアドレステーブルのアドレスを指定し、
NULL までが有効データとなります。

全削除モード時は参照されませんので NULL を指定して下さい。

部品の表示　TKdisplay

機　能

パネルに指定された部品の表示を行ないます。
部品の描画情報の設定を変更した際には、この関数で表示更新を行ないます。
パネルオープン直後は、無意味ですので御注意下さい。

コーリングシーケンス

```
void   TKdisplay   (core, mode, obj) ;

       Ktcore      *core;      /*   パネル管理情報          */
       int         mode;       /*   表示モード              */
       Ktobj       *obj;       /*   表示部品データ          */
```

パラメータ説明

*core　部品表示を行ないたいパネルのパネル管理情報を渡します。

mode　KSINGL　単体表示モード
1部品のみの表示を行ないます。

KARRAY　複数表示モード
部品のアドレステーブルのアドレスをパラメータ
として指定し、複数の部品の表示を行ないます。

KALL　全表示モード
パネルに登録されている全ての部品を再表示します。

*obj　単体表示モード時は、表示部品データのアドレスを指定します。

複数表示モード時は、部品のアドレステーブルのアドレスを指定し、
NULL までが有効データとなります。

全表示モード時は、参照されませんので NULL を指定して下さい。

注　意

既に登録 (TKentry) している部品の再表示 (TKdisplay) と、未登録部品の表示結果は異なる場合があります。

(例)　帳票部品　・・・・・・・・・・　表示しません。
　　ラスター部品　・・・・・・・　スクロールバーは表示しません。

イベント系部品を未登録での表示は控えて下さい。また、部品の位置/大きさが変わる変更は出来ませんので御注意下さい。

表示系部品で位置/大きさが変わる場合、及び、テキスト部品の表示文字が変わる場合には、TKerase関数にて表示消去を行なっておく必要があります。
但し、ボックスカーソル部品、バーカーソル部品、棒グラフ部品は、TKeraseを行なう必要はありません。

部品の消去　TKerase

機　能

パネルに表示されている部品の消去を行ないます。
実際には、部品の描画領域をパネルの色で塗りつぶしを行ないます。部品が重なり合っている場合には、その部分が消えてしまいますので、必要な部分は TKdisplay関数で表示更新を行なう必要があります。

コーリング・シーケンス

```
void   TKerase     (core, mode, obj) ;

       Ktcore      *core;      /*   パネル管理情報          */
       int         mode;       /*   消去モード              */
       Ktobj       *obj;       /*   消去部品データ          */
```

パラメータ説明

*core　部品消去を行ないたいパネルのパネル管理情報を渡します。

mode　KSINGL　単体消去モード
1部品のみの消去を行ないます。

KARRAY　複数消去モード
部品のアドレステーブルのアドレスをパラメータとして指定し、複数の部品の消去を行ないます。

KALL　全消去モード
パネルに登録されている全ての部品を消去します。

*obj　単体消去モード時は、消去部品データのアドレスを指定します。

複数消去モード時は、部品のアドレステーブルのアドレスを指定し、NULL までが有効データとなります。

全消去モード時は、参照されませんので NULL を指定して下さい。

注　意

テキスト部品の場合には、部品に定義している色で塗りつぶします。

イベント待ち　TKevent

機　能

イベント処理を行ないます。
全てのパネルを閉じるか、コールバックルーチンで1以上の値が返されるまで
イベントループ処理を続けます。

コーリング・シーケンス

```
int         TKevent    (root) ;

        Ktroot     *root;          /*   ライブラリルート情報              */
```

戻り値

イベント処理のリターンステータスを示します。

イベント系部品に定義したコールバックルーチンのリターン値が 0 より
大きい値の場合に、その値がイベントステータスとして返されます。

また、タイトルバーのクローズボタンがクリックされ全てのパネルが
閉じた際に、値 1 が返されます。

パラメータ説明

*root　　TKinit 関数 (ライブラリ初期設定) で返された情報を渡します。

注 意

戻り値は、各コールバックルーチンからの戻り値と一致しますが、値 1 は
クローズボタンで使用していますので、アプリケーションでは値 2 以上を使用
する事をお勧めします。

イベントディスパッチ　　TKevdispatch

機　能

Xのイベント情報を解析し、パネルイベント処理を行ないます。
この機能を使用する場合は、ユーザがXのイベントを取得しなければなりません。
また、この関数ではイベントループは行ないません。

イベントマスク

パネルに設定しているイベントマスクは下記の通りです。

KeyPreeeMask ¦KeyReleaseMask ¦ButtonPressMask ¦ButtonReleaseMask ¦
PointerMotionMask ¦EnterWindowMask ¦LeaveWindowMask ¦ExposureMask ¦
FocusChangeMask ¦StructyuerNotifyMask

これ以外のイベントが必要な場合は個々のパネルに対してイベントマスクの値を
再設定して下さい。

注　意

ClientMassageイベントについては注意が必要です。

コーリングシーケンス

```
stat  =  TKevdispatch  (root, event) ;

        Ktroot    *root;     /*  ライブラリ管理情報          */
        XEvent    *event;    /*  Xのイベント構造体           */
        int       stat;      /*  リターンステータス          */
```

戻り値

0	・・・	正常。
上記以外	・・・	コールバックのリターンステータス

パラメータ説明

*root	TKinit 関数で返された情報を渡します。
*event	Xのイベント情報が保存されたアドレスを渡します。

【プログラミング例】

```
main()
{
    Display     *display;                       /*   Xディスプレイ変数      */
    XEvent      event;                          /*   Xイベント構造体        */

    root = (Ktroot *)TKinit();
    if( root != NULL )
    {
        core = (Ktcore *)TKopen( ···· );
        if( core != NULL )
        {
                    :
            display = root->dis;                /*   ディスプレイ変数設定   */

            for( sts=0; sts!=1; )
            {
                XNextEvent( display, &event );  /*   イベント発生待ち       */
                  ユーザ処理
                sts = TKevdispatch( root, &event );  /*   イベント処理          */

                if( root->panel.count==0 )  sts = 1;  /*   パネル全てクローズ？  */
            }
        }
    }
    TKexit( root );
}
```

イベントマスク機能　　TKevmask

機　能

指定したパネルに対するイベントをマスクします。
（イベントを受け取らずに捨てます。）

コーリングシーケンス

```
void   TKevmask   (core, mode, type) ;

       Ktcore     *core;      /*   パネル管理情報          */
       int        mode;       /*   マスクリクエストモード  */
       int        type;       /*   マスクタイプ            */
```

パラメータ説明

*core　　マスク指定パネル情報。
NULL を渡した場合、そのプロセスが管理する全てのパネルとなる。

mode　　マスクリクエスト

ON　・・・　マスクする。
OFF　・・・　マスク解除する。

type　　マスクタイプ

NULL	全てのイベントをマスクする。
EVPOINT	マウス等のポインタイベントをマスクする。
EVKEY	キーボードイベントをマスクする。

EVPOINT と EVKEY の両方ともマスクしたい場合は、OR にて設定出来ます。

例

```
TKevmask   (core, ON, EVPOINT |EVKEY) ;
```

補　足

コールバックルーチンでは、すみやかに処理を終了し、イベント待ちに戻らなければなりませんが、データベース検索処理等の様に処理時間が長い場合があります。この場合、パネルに対して発生したイベントが溜ってしまい、イベントループに戻った時に不用意なイベント処理が実行されてしまいます。これを回避する為に、長い処理を行なう前にイベントマスクを行なう事で不用意なイベントを無視出来ます。

注　意

コールバックルーチンの先頭で、イベントマスクを行なってもコールバックルーチンが呼ばれるまでのわずかな時間の間に発生したイベントにマスクを行なう事は出来ません。

全てのイベントマスクを行なった際には、必ず同じコールバックルーチン内でマスク解除する様にして下さい。二度とイベントを取れなくなります。

メッセージ送信　TKsend

機　能

ウィンドウ間でのメッセージ送信を行ないます。

コーリング・シーケンス

```
int     TKsend      (core, wid, mid, info, len) ;

        Ktcore      *core;      /*    パネル管理情報                    */
        Window      wid;        /*    メッセージ送信先ウィンドウID      */
        int         mid;        /*    メッセージID (ユーザ識別)         */
        char        *info;      /*    関連情報 (メッセージデータ) アドレス */
        int         len;        /*    関連情報データ長                  */
```

パラメータ説明

*core　　送信元のパネル管理情報を渡します。

wid　　送信先パネルのウィンドウIDを示します。

mid　　識別メッセージIDを示します。 (0以上)

*info　　送信データのアドレスを指定します。
データの型は、便宜上char * になっていますが、文字列である必要はありません。
送信データは、**共有メモリ** を介して受け渡しされます。
送信データがない場合は、値 0 を設定します。

len　　データ長 (バイト数) を示します。データがない場合は、値 0 を設定します。

戻り値

0　… 正常送信。
0以外… 送信エラー。エラー内容の詳細は、システム変数の errno を参照ください。
エラーの要因として、「送信先ウインドウが無い」、「共有メモリの不足」、
「送信先ウインドウがイベントキューFullによる受付拒否」が考えられます。

補　足

メッセージを受信するパネルでは、TKnotify関数によりメッセージ受信イベント(KMESG)のコールバック・ルーチンを登録しておく事で、受信処理が行なわれます。

注　意

割り込み処理中 (signal で登録した関数) では、coreを使用する事が出来ませんので、下記に示す、割り込み処理用core情報 (擬似core) を利用する手法を行なって下さい。
また、パネルを持たないプロセスからについてもこの方法で、TKsend関数の利用が可能です。

【割り込み処理用core情報生成手順】

```
static  Ktcore  sig_core;                   /* 割り込み用core定義（実体）*/
main()
{
    sig_core.dis = XOpenDisplay( NULL );    /* Display情報生成           */
    if( sig_core.dis != NULL )
    {
        root = (Ktroot *)TKinit();
          ***************************
          * X-Mate基本プログラミング *
          ***************************
        XCloseDisplay( sig_core.dis );      /* Display情報開放           */
    }
}
```

【割り込み処理手順】

```
割込処理()
{
    TKsend( &sig_core, ···· );
}
```

入力部品状態切り換え　TKkeysw

機　能

入力部品の入力状態の切り換えを行ないます。

コーリングシーケンス

```
int    TKkeysw    (core, obj, mode, line, clum) ;

       Ktcore     *core;     /*   パネル管理情報                  */
       Ktobj      *obj;      /*   入力部品アドレス                */
       int        mode;      /*   キー入力状態                    */
       int        line;      /*   カーソル表示位置(ライン数)      */
       int        clum;      /*   カーソル表示位置(カラム数)      */
```

パラメータ説明

*core　　入力部品が登録されているパネルのパネル管理情報を渡します。

*obj　　状態を変化させる入力部品のアドレスを示します。

mode　　KION　・・・　キー入力オン　：　キー入力を開始します。
　　KIOFF　・・・　キー入力オフ　：　キー入力を終了します。

line
clum　　入力を開始する時のカーソル表示位置を入力エリア内でのライン数、カラム数にて指定します。(0～)
　　入力エリアの最後にカーソルを表示する場合は、ライン数、カラム数共に負の値を指定します。

注　意

例外として帳票部品(Kttable)、新帳票部品(Kttable2)では、
ライン数はセル行番号、カラム数はセル列番号となります。

戻り値

0　… 入力状態の切替え正常。
-1　… 切替えエラー。指定モードにより次の要因が考えられます。
　・KEYON　：既に入力状態にあった部品の入力データが不正です。
　・KEYOFF：指定した部品の入力データが不正です。
その他　… KEYON時に指定した部品の入力前処理が返す戻り値が返ってきます。

● 帳票部品のセル番号は以下の通りです。

	列見出し1	列見出し2	列見出し3	・・・・
行見出し1	セル行番号=0 セル列番号=0	セル行番号=0 セル列番号=1	セル行番号=0 セル列番号=2	・・・・
行見出し2	セル行番号=1 セル列番号=0	セル行番号=1 セル列番号=1	セル行番号=1 セル列番号=2	・・・・
：	：	：	：	

● 新帳票部品のセル番号は以下の通りです。

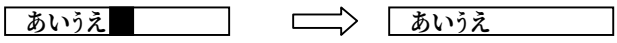
セル行番号=0 セル列番号=0	セル行番号=0 セル列番号=1	セル行番号=0 セル列番号=2	・・・・
セル行番号=1 セル列番号=0	セル行番号=1 セル列番号=1	セル行番号=1 セル列番号=2	・・・・
：	：	：	

・データ入力部品(Ktidata)等で入力行が一行しかない場合は、ライン数の設定は無意味となります。
・切り換えを行なう部品は、登録(TKentry)されている必要があります。

パネル内入力受付状態解除　　TKkeyoff

機　能

キー入力受付状態の部品がある時、それをキー入力無効状態にします。



（キー入力受付状態）　　（キー入力無効状態）

コーリングシーケンス

```
int     TKkeyoff    (core) ;
        Ktcore      *core;          /*    パネル管理情報                */
```

パラメータ説明

*core　　パネル管理情報を渡します。

戻り値

0　… キー入力受付解除が正常に行われたことを示します。
-1　… 入力中部品の入力データが不正です。

入力モード切り換え　TKimode

機　能

入力機能(挿入/上書き状態)の切り換えを行ないます。
入力中は、状態の即時切り換えを行ないます。
非入力中は、次回入力開始時の状態となります。

コーリングシーケンス

```
void  TKimode   (mode) ;
      int       mode;       /*  入力モード                  */
```

パラメータ説明

mode　KINS　インサート(押入)モード

キー入力状態をインサートモード(Ins キーを押した状態と同じ)にします。

KREP　リプレース(上書き)モード

キー入力状態をリプレースモード(上書き状態)にします。

コピー&ペースト機能制御　　TKpastlock

機　能

キー入力部品に表示している文字をマウス操作でコピー及びペーストする機能を設定します。

コーリングシーケンス

```
void   TKpastlock  (mode) ;
         int          mode;          /*    ペーストモード                    */
```

パラメータ説明

mode	KCLOCK	コピー禁止 文字の写し取り(コピー)機能を無効にします。
	KPLOCK	ペースト禁止 写し取った文字の貼り込み(ペースト)機能を 無効にします。
	KLOCK	コピー&ペースト禁止 コピー及びペースト機能共に禁止します。
	KUNLOCK	禁止状態解除 全ての禁止状態を解除し、機能を有効にします。

補　足

旧仕様で定義されていた、ON は KLOCK、OFF は KUNLOCK と等価です。

コピー文字情報取得　　TKgetbuff

機　能

マウス操作で、写し取った文字情報を取得します。

コーリングシーケンス

```
char   *TKgetbuff  (core) ;
        Ktcore      *core;          /*    パネル管理情報                */
```

戻り値

写し取った文字列のアドレスを返します。
写し取られていない場合は、NULL を返します。
このアドレスは使用後、free関数で開放して下さい。

パラメータ説明

*core　　パネル管理情報を渡します。

帳票セルデータ更新　　TKchgcell

機　能

セル行番号、セル列番号で指定されたセル情報を更新します。
セルの設定及び、データの変更を行なった場合に使用します。
TKdisplay関数でも同様の機能を行ないますが、変更セル数が少ない場合には、
この関数を使用する事により処理の負担が軽減します。特にセル数の多い帳票
ほど有効です。

コーリングシーケンス

```
void   TKchgcell   (core, obj, line, clum) ;

       Ktcore     *core;      /*   パネル管理情報          */
       Ktobj      *obj;       /*   帳票部品アドレス        */
       int        line;       /*   セル行番号              */
       int        clum;       /*   セル列番号              */
```

パラメータ説明

*core　　TKopen 関数で返される情報を渡します。

*obj　　帳票/新帳票部品のアドレスを示します。

line　　帳票のセル番号 (行/列) を示します。
clum　　-1を指定すると、その行または列のセルを一括更新します。

セル番号は次の通りです。

● 帳票部品の場合

	列見出し1	列見出し2	列見出し3	・・・・
行見出し1	セル行番号=0 セル列番号=0	セル行番号=0 セル列番号=1	セル行番号=0 セル列番号=2	・・・・
行見出し2	セル行番号=1 セル列番号=0	セル行番号=1 セル列番号=1	セル行番号=1 セル列番号=2	・・・・
：	：	：	：	

● 新帳票部品の場合

セル行番号=0 セル列番号=0	セル行番号=0 セル列番号=1	セル行番号=0 セル列番号=2	・・・・
セル行番号=1 セル列番号=0	セル行番号=1 セル列番号=1	セル行番号=1 セル列番号=2	・・・・
：	：	：	

マウスカーソルの移動　TKwarp

機　能

指定したパネルに対して、マウスカーソルを移動します。

コーリングシーケンス

```
void   TKwarp     (core, wid, x, y) ;

       Ktcore     *core;     /*   パネル管理情報                  */
       Window     wid;       /*   マウスカーソルを移動したい
                                      パネルのウィンドウID        */
       int        x;         /*   マウスカーソル表示位置  X座標   */
       int        y;         /*   マウスカーソル表示位置  Y座標   */
```

パラメータ説明

*core	パネル管理情報を渡します。
wid	マウスカーソルを表示したいパネルのウィンドウIDを示します。 自ウィンドウに指定する時は、core->kit と設定します。
x y	マウスカーソルを表示する位置 (XY座標) を示します。 座標位置は指定のパネルを基準とします。

マウスカーソル変更　TKchgcursor

機　能

指定のパネル上でのマウスカーソルの形状を変更します。
形状の種類は、Xカーソルです。

コーリングシーケンス

```
void   TKchgcursor      (core, cid) ;

            Ktcore      *core;      /*   パネル管理情報          */
            int         cid;        /*   カーソルID              */
```

パラメータ説明

*core	変更を行ないたいパネルのパネル管理情報を渡します。
cid	カーソルの種類はXが準備している cursorfont.h 内で定義された IDを用いて指定します。

注 意

イベントループ(TKevent)の前では実行出来ません。何らかのイベント処理内で実行して下さい。

ビットマップカーソル　TKchgbcursor

機　能

指定のパネル上でのマウスカーソルを任意のビットマップ形状に変更します。

コーリングシーケンス

```
void   TKchgbcursor     (core, fb, mb, fc, bc) ;

        Ktcore      *core;      /*   パネル管理情報              */
        Ktbitmap    *fb;        /*   カーソル用ビットマップ情報  */
        Ktbitmap    *mb;        /*   カーソルマスク用ビットマップ情報 */
        char        *fc;        /*   カーソル表示色              */
        char        *bc;        /*   カーソルマスク表示色        */
```

パラメータ説明

*core	変更を行ないたいパネルのパネル管理情報を渡します。
*fb	表示するカーソルのビットマップ情報を示します。
*mb	表示するカーソルのマスク用ビットマップ情報を示します。
*fc	カーソルの表示色を指定します。 NULL が指定された場合は黒色となります。
*bc	カーソルマスクの表示色を指定します。 NULL が指定された場合は白色となります。

構 造 体

```
typedef struct
{
    int         type;       /*   ビットマップタイプ              */
    int         w;          /*   ビットマップ幅                  */
    int         h;          /*   ビットマップ高さ                */
    int         hx;         /*   ホットスポット  X               */
    int         hy;         /*   ホットスポット  Y               */
    char        *data;      /*   ファイル名又は、メモリデータ    */
}   Ktbitmap;
```

メンバー説明

type	KBFILE	・・・・・・ ファイルデータ
	KBDATA	・・・・・・ メモリデータ
w h	実際のビットマップイメージの大きさを指定します。 ビットマップタイプがメモリデータの時のみ有効です。	
hx hy	カーソルポジションを指定します。 ビットマップタイプがメモリデータの時のみ有効です。	
*data	ファイルデータの場合　：　ビットマップファイルのパス名を示します。 メモリデータの場合　：　ビットマップデータの保存アドレスを示します。	

注 意

イベントループ（TKevent）の前では実行出来ません。何らかのイベント処理内で実行して下さい。

メニュー部品のカーソル指定　TKmenucursor

機　能

メニュー部品のカーソルの指定を行ないます。
指定可能な形状は、Xカーソルです。

コーリングシーケンス

```
TKmenucursor    (id) ;
        int         id;         /*  指定フォントカーソルID          */
```

パラメータ説明

id	カーソルID。 XフォントカーソルのカーソルIDを指定します。

例

左上矢印を指定したい場合、XC_top_left_arrow を指定します。

子ウィンドウ削除　TKkill

機　能

　指定した子ウィンドウを削除します。
そのウィンドウに対してさらに子として親子関係を持っているウィンドウがある場合には、
同時に削除します。

コーリングシーケンス

```
void    TKkill  (core, wid) ;

        Ktcore  *core;      /*  パネル管理情報          */
        Window  wid;        /*  削除したい子ウィンドウのID  */
```

パラメータ説明

*core	パネル管理情報を渡します。
wid	削除したい子ウィンドウのウィンドウIDを指定します。

全子ウィンドウの削除　TKchildkill

機　能

指定されたパネルに関係している全ての子ウィンドウを削除します。

コーリングシーケンス

```
void    TKchildkill    (core) ;
        Ktcore    *core;        /*   パネル管理情報              */
```

パラメータ説明

*core　削除したいパネルのパネル管理情報を渡します。

パネル設定変更　TKchgpanel

機　能

パネルの設定を変更します。

コーリングシーケンス

```
void   TKchgpanel      (core, panel) ;

           Ktcore      *core;      /*   パネル管理情報          */
           Ktpanel     *panel;     /*   パネル定義情報          */
```

パラメータ説明

*core　　変更を行ないたいパネルのパネル管理情報を渡します。

*panel　　パネル定義情報のアドレスを指定します。

注　意

(1)　パネルタイプの変更は出来ません。

例:　通常パネル(KPROC)　→　システムダイアログパネル(KSDLG)

(2)　パネルの状態が、アイコン及びアンマップ状態の際には機能しません。

(3)　マウスカーソルの変更を行なっている場合には、初期状態に戻ってしまいます。
これを回避するには、パネル定義情報のメンバ cid に -1 を設定して下さい。

フォント種別追加　TKaddfont

機　能

デフォルトで用意しているフォントの他に、新たにフォントを追加します。

追加の際には、ANKフォントと日本語フォントをペアで指定します。
この時、日本語フォントの幅はANKフォントの幅の2倍になる様にして下さい。
また、プロポーショナル（可変幅）フォントは使用せず、固定幅フォントを使用して下さい。

この関数のリターン値を部品のフォント種別に設定する事により、追加したフォントを
使用して文字を表示する事が出来ます。

次のコマンドを入力すると、Xで定義されているフォント名称が表示されます。

```
%    xlsfonts  Return
```

コーリングシーケンス

```
int    TKaddfont   (root, afont, jfont) ;

       Ktroot      *root;      /*   パネル管理情報          */
       char        *afont;     /*   ANKフォント名称         */
       char        *jfont;     /*   日本語フォント名称      */
```

戻り値

フォントのID番号を示します。
負の値の場合は、設定エラーです。
このIDを部品のフォント種別に指定します。

パラメータ説明

*root　パネル管理情報を渡します。

*afont　Xで定義されている ANK文字のフォント名称（文字列）指定します。
ANK文字を使用しない時は NULL を指定します。

次のコマンドを入力すると、Xで定義されている半角カタカナ入りの
ANKフォント名称が表示されます。

```
%    xlsfonts  ¦ grep  jisx0201    Return
```

*jfont　Xで定義されている日本語文字のフォント名称（文字列）を指定します。
日本語文字を使用しない時は NULL を指定します。

次のコマンドを入力すると、Xで定義されている日本語フォント名称が
表示されます。

```
%    xlsfonts  ¦ grep  jisx0208    Return
```

参 考

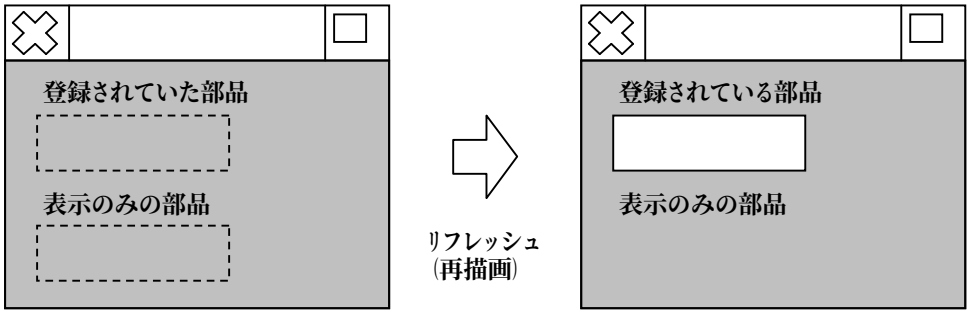
次のコマンドを入力すると、指定したフォント文字を見る事が出来ます。

```
%    xfd  -fn    フォント名称   Return
```

パネルクリア　TKclear

機　能

パネル全体をパネル背景色で塗りつぶします。
パネルに部品が登録（TKentry）されている場合は、再描画を行なうと部品が再度表示されます。
パネルに部品が表示（TKdisplay）のみで描画されていた場合は、再描画を行なっても部品は表示されません。



TKclear　を行なった直後　　登録されていた部品は表示されます

コーリングシーケンス

```
void  TKclear  (core);
        Ktcore  *core;      /*  パネル管理情報  */
```

パラメータ説明

*core　クリアを行ないたいパネルのパネル管理情報を渡します。

キャンバスcore抽出　　TKcanvcore

機　能

キャンバス部品の描画パネルのパネル管理情報(core)を取得します。
キャンバス部品は登録(TKentry)されている必要があります。

コーリングシーケンス

```
Ktcore      *TKcanvcore   (canv) ;
        Ktcanv      *canv;          /*   キャンバス部品                */
```

戻り値

描画パネルのパネル管理情報。
キャンバス内に描画等の操作に使用するパネル管理情報を示します。
キャンバス部品が未登録の場合は NULL が返ります。

パラメータ説明

*canv　　キャンバス部品のアドレスを示します。

注　意

この関数で取得したcore情報は、キャンバスに部品を描画する為にのみ使用して下さい。また、このcore情報で TKclose は行なわないで下さい。

子プロセス起動　　CWexec

機　能

子プロセスを起動します。(子プロセス起動簡易処理)

コーリングシーケンス

```
int     CWexec      (path, wid, p1, p2, p3, p4, p5) ;

        char        *path;      /*   子プロセスパス名称アドレス      */
        Window      wid;        /*   自ウィンドウID                  */
        int         p1;         /*   データ    1                     */
        int         p2;         /*   データ    2                     */
        int         p3;         /*   データ    3                     */
        int         p4;         /*   データ    4                     */
        int         p5;         /*   データ    5                     */
```

戻り値

子プロセスのプロセスIDを返します。
値 -1 は起動出来なかった事を示します。

パラメータ説明

*path　　起動したいプロセスのファイル名称を指定します。

wid　　子ウィンドウが起動する時、親となるウィンドウのIDです。
自ウィンドウIDの値は、パネル管理情報(core)内の kit に格納しています。

このパラメータは次に続く p1～p5 に対して同等の意味ではあるのですが、
必ずしも'自ウィンドウID' の設定を行なう必要はありません。しかしながら、
子プロセスは起動時に親のウィンドウIDを必要としますので、ほとんどの場合は
この設定を必要とします。

p1
p2
p3
p4
p5　　p1～p5 までは、子プロセスのウィンドウに渡すデータを設定します。
パラメータにデータを設定しない場合は NULL を設定します。

例えば、 p1 から p4 までに起動プロセスのウィンドウの座標及び大きさを設定し、 p5 はどの親ウィンドウから起動されたかを判別するIDに使用する。
(CWgetarg関数参照)

補　足

この関数は、外部プログラムを起動する簡易機能です。
fork関数とexeclp関数を直接利用頂いても構いません。

プロセス起動情報取得　　CWgetarg

機　能

親プロセスから送られる情報を取り込みます。
CWexec関数にて起動された場合に使用します。

コーリングシーケンス

```
int     CWgetarg    (argc, argv, &wid, &p1, &p2, &p3, &p4, &p5) ;

        int         argc;       /*  引数の数                    */
        char        *argv[ ];   /*  引数のポインタ配列          */
        Window      wid;        /*  親ウィンドウID              */
        int         p1;         /*  データ    1                 */
        int         p2;         /*  データ    2                 */
        int         p3;         /*  データ    3                 */
        int         p4;         /*  データ    4                 */
        int         p5;         /*  データ    5                 */
```

戻り値

```
        int         count;      /*  パラメータ数                */
```

パラメータ説明

argc	標準入力からのパラメータ数を格納します。
*argv[ ]	標準入力からのパラメータデータアドレスを示します。
wid	パラメータを保存するアドレスを示します。
p1	
p2	
p3	
p4	
p5	

親/子プロセス(CWexec / CWgetarg) プログラム例

CWexec と CWgetarg を使ったプログラム例を以下に示します。

【親プロセス処理】

① 子プロセスのウィンドウの表示位置をパラメータとしてセットします。

② CWexecにより子プロセスを起動します。

③ コーディング

```
CWexec( "child", core->kit, 200, 100, 500, 300, 1);
```

起動プロセス パス名称アドレス　　自ウィンドウID　　p1　p2　p3　p4　p5

【子プロセス処理】モジュールファイル名 = child

```
    Ktpanel     panel;
    Window      bf_pwid;

main (argc, argv)
    int     argc;
    char   *argv[ ];
{
    Window      wid;
    int         x, y, w, h, data;

    bf_pwid = NULL;         /*    親ウィンドウID 初期値 NULL セット    */
    count = CWgetarg (argc, argv, &wid, &x, &y, &w, &h, &data);
    if ( count > 1 )        /*    親ウィンドウ起動チェック             */
    {
        bf_pwid  =  wid;    /*    親ウィンドウID セット                */
        panel.wx =  x;      /*    ウィンドウ情報セット                 */
        panel.wy =  y;      /*    X軸200、Y軸100、幅500、高さ300       */
        panel.ww =  w;      /*    のパネル情報を設定                   */
        panel.wh =  h;
    }
    root = (Ktroot *) TKinit ();
    if ( root != NULL)
    {                       /*    パネルオープン                       */
        core = (Ktcore *) TKopen (root, bf_pwid, &panel);
            :
```

Tkopenのパラメータとして、親ウィンドウIDとパネル定義情報のアドレスを設定します。
これによりウィンドウの親子関係をつくり、親ウィンドウが指定した位置にウィンドウをオープンしています。

マウスカーソル変更2　　TKchgcursor2

機　能

　　指定のパネルのマウスカーソルの形状を任意の Xカーソル に変更します。
同時に、カーソルの色を指定する事が可能です。

コーリングシーケンス

```
void   TKchgcursor2      (core, id, fc, bc) ;

       Ktcore    *core;      /*   パネル管理情報          */
       int       id;         /*   Xフォントカーソル ID    */
       char      *fc;        /*   カーソル色              */
       char      *bc;        /*   カーソルマスク色        */
```

パラメータ説明

*core	カーソル変更を行ないたいパネルのパネル管理情報を渡します。
id	カーソル種別はXが準備している cursorfont.h で定義された ID を用いて指定します。
*fc	カーソルの表示色を指定します。
*bc	カーソルマスクの表示色を指定します。

注　意

　　パネルオープン直後では実行出来ません。何らかのイベント処理内で行なって下さい。

ワイルドカード関数　TKwildchk

機　能

文字列を任意のワイルドカード書式と比較し、文字列の真偽を行ないます。

コーリングシーケンス

```
int     TKwildchk  (text, wild) ;

        char    *test;          /*   チェック文字列              */
        char    *wild;          /*   ワイルドカード文字列        */
```

戻り値

比較OKなら 0、NGなら -1 を返します。

パラメータ説明

*text　　比較を行なう文字列を指定します。

*wild　　ワイルドカード書式の文字列を指定します。
書式は次の通りです。

```
*  :  文字列
?  :  1文字
{ } :  文字列指定 (複数指定可。セパレータは',')
[ ] :  1文字指定  (範囲指定可。ex:[0-5])
```

【例 1】

```
   "{abc, xyz}[0-9][0-9].c"
の場合
   "abc00.c"   ・・・  OK
   "abc89.c"   ・・・  OK
   "abc001.c"  ・・・  NG
   "xyz56.c"   ・・・  OK
   "def00.c"   ・・・  NG
```

【例 2】

```
   "*a?c*"
の場合
   "abc"       ・・・  OK
   "zabcdef"   ・・・  OK
   "00abbc99"  ・・・  NG
```

ラベル登録関数　TKlabelent

機　能

　ラベル名（文字列）とその部品アドレスを登録します。
同一のラベル名が登録された際には、後登録が有効となります。
　表計算を行なう際に、このラベル登録を行なっておく事で、式中に指定したラベルへの参照を促します。

コーリングシーケンス

```
void   TKlabelent  (root, label) ;

       Ktroot      *root;        /*   root情報                    */
       Ktlabel     *label;       /*   ラベル情報                  */
```

パラメータ説明

*root　　TKinit 関数で返された情報を渡します。

*label　　ラベル名と部品アドレスを定義した、ラベル情報を格納したアドレス。
ラベル情報は、ラベルアドレスが NULL までの配列情報。

ラベル構造体

```
typedef struct
{
   char        *label;        /*   ラベル名（64文字以内）      */
   Ktobj       *obj;          /*   部品アドレス                */
}  Ktlabel;
```

《例》

```
Ktlabel labels[ ] =
{
   {  "data3",     &data3.hed     },
   {  "table2",    &table2.hed    },
   {  NULL,        NULL           }
};
```

ラベル削除関数　TKlabeldel

機　能

ラベル登録関数(TKlabelent)により登録を行なったラベルを削除します。

コーリングシーケンス

```
void   TKlabeldel   (root, label) ;

       Ktroot      *root;        /*    root情報                    */
       Ktlabel     *label;       /*    ラベル情報                  */
```

パラメータ説明

*root	TKinit 関数で返された情報を渡します。
*label	ラベル名と部品アドレスを定義した、ラベル情報を格納したアドレス。 ラベル情報は、ラベルアドレスが NULL までの配列情報。

入力位置取得関数　　TKkeyposi

機　能

指定部品が入力中の場合、キーカーソルの位置を取得します。
入力中でない場合には、カーソル位置の情報として -1 が返されます。

コーリングシーケンス

```
int     TKkeyposi   (core, obj, cp, lp, line, clum) ;

        Ktcore  *core;      /*  パネル管理情報              */
        Ktobj   *obj;       /*  入力部品                    */
        int     *cp;        /*  桁位置                      */
        int     *lp;        /*  行位置                      */
        int     *line;      /*  行番号 (帳票の場合)         */
        int     *clum;      /*  列番号 (帳票の場合)         */
```

戻り値

0　:　正常終了
-1　:　パラメータエラー (指定部品間違い)

パラメータ説明

*core　　TKopen 関数で返された情報を渡します。

*obj　　調査を行なう入力部品を指定します。

*cp　　入力カーソル位置
*lp　　指定の部品が入力中であれば、入力位置を返します。
　　非入力中の場合には、cp に -1 を返します。

*line　　入力セル位置
*clum　　入力中のセル位置を返します。指定の部品が帳票部品のみ有効。

※ lp, line, clum の戻り値が不要な場合には、NULL を指定する事も可

入力状態取得関数　TKfepmode

機　能

現在の入力/非入力状態を取得します。
入力状態の場合には、通常入力中/日本語入力中を通知します。

コーリングシーケンス

```
int     TKfepmode (core);

        Ktcore  *core;          /*  パネル管理情報          */
```

戻り値

-1　:　非入力中
0　:　入力中(非日本語)
1　:　日本語入力中

パラメータ説明

*core　TKopen 関数で返された情報を渡します。

スクロールバー部品スクロール量変更　TKsbarinc

機　能

スクロールバー部品のスクロール量を変更します。
Motifライク（環境変数 KTMOTIF=on を設定）において、スクロールバーのアローボタン（矢印）を操作すると、デフォルトで1（ドット）ずつ移動します。
この移動量を任意の量に変更します。

コーリングシーケンス

```
void   TKsbarinc ( *obj,  count ) ;

   Ktsbar      *obj;          /*   スクロール部品情報          */
   int         count;         /*   移動量                      */
```

戻り値

なし

パラメータ説明

*obj	移動量の変更を行なうスクロールバー部品のアドレスを示す。
count	移動量（ドット数）を指定します。

リスト部品スクロール量変更　　TKlistinc

機　能

ラスター部品のスクロール量を変更します。
Motifライク（環境変数 KTMOTIF=on を設定）において、リスト部品のスクロールバーのアローボタン（矢印）を操作すると、デフォルトで1（行）ずつ移動します。
この移動量を任意の量に変更します。

コーリングシーケンス

```
void   Tklistinc ( obj,  count ) ;

   Ktlist      *obj;          /*   リスト部品情報          */
   int         count;         /*   移動量                  */
```

戻り値

なし

パラメータ説明

*obj	移動量の変更を行なうリスト部品のアドレスを示す。
count	移動量（行数）を指定します。

ラスター部品スクロール量変更　　TKrastinc

機　能

ラスター部品のスクロール量を変更します。
Motifライク（環境変数 KTMOTIF=on を設定）において、ラスター部品のスクロールバーのアローボタン（矢印）を操作すると、デフォルトで1（ドット）ずつ移動します。
この移動量を任意の量に変更します。

コーリングシーケンス

```
void   TKrastinc ( obj,  w,  h ) ;

    Ktrast      *obj;          /*   ラスター部品情報          */
    int         w;             /*   横移動量                  */
    int         h;             /*   縦移動量                  */
```

戻り値

なし

パラメータ説明

*obj	移動量の変更を行なうラスター部品のアドレスを示す。
w	横移動量（ドット数）を指定します。
h	縦移動量（ドット数）を指定します。

キャンバス部品スクロール量変更　　TKcanvinc

機　能

キャンバス部品のスクロール量を変更します。
Motifライク（環境変数 KTMOTIF=on を設定）において、キャンバス部品のスクロールバーのアローボタン（矢印）を操作すると、デフォルトで1（ドット）ずつ移動します。
この移動量を任意の量に変更します。

コーリングシーケンス

```
void   TKcanvinc ( obj,  w,  h ) ;

    Ktcanv      *obj;          /*   キャンバス部品情報          */
    int         w;             /*   横移動量                    */
    int         h;             /*   縦移動量                    */
```

戻り値

なし

パラメータ説明

*obj	移動量の変更を行なうキャンバス部品のアドレスを示す。
w	横移動量（ドット数）を指定します。
h	縦移動量（ドット数）を指定します。

帳票部品スクロール量変更　TKtablinc

機　能

帳票部品のスクロール量を変更します。
Motifライク（環境変数 KTMOTIF=on を設定）において、帳票部品のスクロールバーのアローボタン（矢印）を操作すると、デフォルトで1（方眼）ずつ移動します。
この移動量を任意の量に変更します。

コーリングシーケンス

```
void   TKtablinc ( obj,  w,  h ) ;

    Kttable     *obj;          /*   帳票部品情報          */
    int         w;             /*   横移動量              */
    int         h;             /*   縦移動量              */
```

戻り値

なし

パラメータ説明

*obj	移動量の変更を行なう帳票部品のアドレスを示す。
w	横移動量（方眼数）を指定します。
h	縦移動量（方眼数）を指定します。

新帳票部品スクロール量変更　　TKtbl2inc

機　能

新帳票部品のスクロール量を変更します。
Motifライク（環境変数 KTMOTIF=on を設定）において、新帳票部品のスクロールバーのアローボタン（矢印）を操作すると、デフォルトで1（方眼）ずつ移動します。
この移動量を任意の量に変更します。

コーリングシーケンス

```
void   TKtbl2inc ( obj,  w,  h ) ;

    Kttable2      *obj;        /*   新帳票部品情報          */
    int           w;           /*   横移動量                */
    int           h;           /*   縦移動量                */
```

戻り値

なし

パラメータ説明

*obj	移動量の変更を行なう新帳票部品のアドレスを示す。
w	横移動量（方眼数）を指定します。
h	縦移動量（方眼数）を指定します。

新テキスト部品スクロール量変更　　TKntextinc

機　能

新テキスト部品のスクロール量を変更します。
Motifライク（環境変数 KTMOTIF=on を設定）において、新テキスト部品のスクロールバーのアローボタン（矢印）を操作すると、デフォルトで1（桁/行）ずつ移動します。
この移動量を任意の量に変更します。

コーリングシーケンス

```
void   TKntextinc ( obj,  w,  h ) ;

    Ktntext     *obj;       /*   新テキスト部品情報          */
    int         w;          /*   横移動量                    */
    int         h;          /*   縦移動量                    */
```

戻り値

なし

パラメータ説明

***obj**	移動量の変更を行なうス新テキスト部品のアドレスを示す。
w	横移動量（桁数）を指定します。
h	縦移動量（行数数）を指定します。

4 定義データ タイプ情報

X-Mate開発ライブラリの各データ情報のタイプ、定義データの種類を以下に示します。

① オブジェクトタイプ情報

② 各オブジェクトモード定義情報

③ マウスボタンタイプ情報

④ フォントタイプ情報

⑤ カラーコード情報

⑥ 部品操作関数モード情報

⑦ テキスト入力状態切り換え関数　状態フラグ

⑧ アイコン状態制御関数　制御モード

⑨ パネル状態制御関数　制御モード

⑩ ビットマップカーソル関数　ビットマップタイプ

⑪ グラフィックコンテキスト(GC) ファンクション

① オブジェクトタイプ定義

部品(オブジェクト)名	値
テキスト	KOTEXT
リピートテキスト	KORTEXT
ライン	KOLINE
罫線	KOKEI
矩形	KORECT
円弧	KOARC
ポリゴン	KOPOLY
ビットマップ	KOIMAG
棒グラフ	KOBARG
バーカーソル	KOBARC
ボックスカーソル	KOBOXC
ラスター	KORAST
キャンバス	KOCANV
ボタン	KOBUTN
テキスト入力	KOITEXT
新テキスト	KONTEXT
データ入力	KOIDATA
データ3	KODATA3
スクロールバー	KOSBAR
ポインタ	KOPOIT
ムーブ	KOMOVE
プルダウンメニュー	KOMENU
新メニュー	KONMENU
リスト	KOLIST
帳票	KOTABLE
新帳票	KOTABLE2
複合部品	KOCOMB

② 各オブジェクトモード定義

テキスト　　リピートテキスト

モード	値
重ね書き	KTOR
上書き	KTSET
反転	KTREV
横書き	KTANG

ライン

ラインモード	値
継続出力	KLMODE
単体出力	KLSING
グラフ表示出力	KLGRPH

座標モード	値
絶対座標	KLORG
相対座標	KLPRV

注 意　　継続出力モード以外は相対座標は使えません。

罫線

罫線モード	値
横　一定間隔	KKYOKO
縦　一定間隔	KKTATE
横　可変間隔	KKYOKOV
縦　可変間隔	KKTATEV

矩形

矩形モード	値
塗りつぶし	KRBAR
枠	KRWAK
枠付き塗りつぶし	KRBOX
影、枠付き塗りつぶし	KRFLW
UPボタン	KRUPB
DOWNボタン	KRDWB

円弧

円弧モード	値
塗りつぶし	KAPNT
枠	KAWAK
枠付き塗りつぶし	KACMB

ファイルタイプ	値
月形	KCHOR
扇形	KSLIC

ポリゴン　(多角形)

ポリゴンモード	値
塗りつぶし	KPPNT
枠	KPWAK
枠付き塗りつぶし	KPCMB

ビットマップ

ビットマップモード	値
イメージ　ファイルデータ	KIFIL
イメージ　メモリデータ	KIDAT

バーカーソル　　スクロールバー　　棒グラフ

カーソルモード	値
縦	KBTATE
横	KBYOKO

(スクロールバーのみ)

カーソルモード	値
縦　モード2	KBTATE2
横　モード2	KBYOKO2
バーボタン	KBBT

ボックスカーソル

カーソルモード	値
枠	KCBOX
塗りつぶし	KCREC
円	KCARC

ラスター

ラスターモード	値
スクロールバーなし	KRNORM
スクロールバーあり	KRSBAR

キャンバス

キャンバスモード	値
スクロールバーなし	KCNORM
スクロールバーあり	KCSBAR

ボタン

ボタンモード	値
ボタンモーメンタリ	KBMOM
ボタンオルタネート	KBALT

テキスト入力

入力モード	値
枠なし	KINOM
枠付き	KIWAK
罫線付き	KIKEI

新テキスト　データ入力　　データ3

状態フラグ	値
入力中	KION
非入力中	KIOFF

入力モード	値
入力あり	KION
表示のみ	KIOFF

(新テキストのみ)

表示モード	値
枠付き	KITW
罫線付き	KITK
縦スクロール	KITVS
横スクロール	KITHS

データ入力　データ3

表示モード	値
下線	KIDINL
枠	KIDINR
UPパネル	KIDINU
DOWNパネル	KIDIND

新メニュー

モード	値
プルダウンメニュー	KMPUL
メニューボタン（▽マークあり）	KMBTN
メニューボタン（▽マークなし）	KMBTN2
ポップアップメニュー	KMPOP

アイテムタイプ	値
テキストアイテム	KMTEXT
ビットマップアイテム	KMBMAP
サブメニュー	KMSUB
見出し	KMTITL
次列	KMNEXT
ブランク	KMNOP
区切り	KMSEP
定義情報の終了	NULL

アイテムフラグ	値
有効	ON
無効	OFF

帳票　新帳票　データ入力　データ3　リスト（表示位置）

表示位置モード	値
右詰め	KPR
中央	KPM
左詰め	KPL

新帳票　データ3　（表示位置）

表示位置モード	値
上詰め	KTBUP
中央	KTBMI
下詰め	KTBDW

帳票　新帳票　（データモード）

データモード	値
表示のみ	KIDDSP
入力あり	KIDIN

データ3　　新帳票　（データ型）

データ型	値
10進数	KDDEC
16進数	KDHEX
8進数	KDOCT
符号なし10進数	KDUDEC
符号なし16進数	KDUHEX
符号なし8進数	KDUOCT
10進数（short)	KDSDEC
16進数（short)	KDSHEX
8進数（short)	KDSOCT
符号なし10進数（short)	KDUSDEC
符号なし16進数（short)	KDUSHEX
符号なし8進数（short)	KDUSOCT
小数点表示（float)	KDFLOAT
指数表示（float)	KDEXP
上記2つの短い方	KDGFORM
小数点表示（double)	KDDOUBL
指数表示（double)	KDDEXP
上記2つの短い方	KDDGFRM
数字	KDNUM
アスキー　（日本語不可）	KDANK
テキスト　（日本語可）	KDTXT
パスワード入力	KDPWD

帳票　　データ入力　　（データ型）

データ型	値
10進数	KDDEC
16進数	KDHEX
8進数	KDOCT
小数点表示	KDFLOAT
指数表示	KDEXP
gタイプ	KDGFORM
数字	KDNUM
アスキー　（日本語不可）	KDANK
テキスト　（日本語可）	KDTXT

リスト

モード	値
罫線付き	KLK
枠付き	KLW
スクロールバー付き	KLS
複数選択	KLF

未対応

③　マウスボタン　マスクタイプ

指定タイプ	値
左ボタン使用	Button1Mask
中央ボタン使用	Button2Mask
右ボタン使用	Button3Mask
全ボタン有効	Button123Mask

④　フォントタイプ

1.

文字データ種別	値
スモールタイプ	SKANA
ミドルタイプ	MKANA
ラージタイプ	LKANA

2.

ビットマップデータ種別	値
ビットマップファイルシンボル	KFIMF
ビットマップデータシンボル	KFIMD

⑤　カラー指定

TKdef.hファイルに以下の色がデファインで定義されています。

表 示 色	値	R G B 値
ブラック	BLACK	#000000000
ホワイト	WHITE	#255255255
ダークグレイ	DGRAY	#084084084
ライトグレイ	LGRAY	#168168168
グレイ	GRAY	#192192192
レッド	RED	#255000000
グリーン	GREEN	#035147035
ブルー	BLUE	#000000255
イエロー	YELLOW	#255255000
シアン	CYAN	#000255255
マジェンタ	MAGENTA	#255000255
ピンク	PINK	#216191216
パールグリーン	PGREEN	#143188143
ライトブルー	LBLUE	#191216216
ウィート	WHEAT	#216216191
背景色	BACK	――――

上記以外の色を使用する場合は、次の形式で指定します。

"#RRRGGGBBB"

RRR ・・・ 赤色の割り合い (0 ～ 255)
GGG ・・・ 緑色の割り合い (0 ～ 255)
BBB ・・・ 青色の割り合い (0 ～ 255)

また、Xが準備しているカラーデータベース、rgb.txtで定義している文字列も使用する事が出来ます。

⑥　部品操作関数の各パラメータで定義

登録/削除/表示/消去　　モード

モード	値
単体モード	KSINGL
複数モード	KARRAY
全体モード	KALL

⑦ 入力状態切り換えサブルーチン
TKkeysw関数　　状態フラグ

キー入力状態	値
キー入力ON	KION
キー入力OFF	KIOFF

⑧ アイコン状態制御サブルーチン
TKiconning関数　　制御モード

モード	値
アイコン化	ON
非アイコン化	OFF

⑨ パネル状態制御サブルーチン
TKmapping関数　制御モード

モード	値
可視化（マップ）	ON
不可視化（アンマップ）	OFF

⑩ ビットマップカーソルサブルーチン
TKchgbcursor関数　ビットマップタイプ

データ種別	値
ファイルデータ	KBFILE
メモリデータ	KBDATA

⑪ グラフィックコンテキスト（GC）ファンクション

描画ファンクションとして以下のものがあります。

関数名称	オペレーション
GXclear	0
GXand	src AND dst
GXandReverse	src AND NOT dst
GXcopy	src
GXandInverted	（NOT src）AND dst
GXnoop	dst
GXxor	src XOR dst
GXor	src OR dst
GXnor	（NOT src）AND（NOT dst）
GXequiv	（NOT src）XOR dst
GXinvert	NOT dst
GXorReverse	src OR（NOT dst）
GXcopyInverted	NOT src
GXorInverted	（NOT src）OR dst
GXnand	（NOT src）OR（NOT dst）
GXset	1

※（GXcopy はグラフィックスコンテキスト内の function コンポーネントデフォルト値です。）

5 環境変数情報

X-Mateを使用して作成したアプリケーションの実行時に、環境変数の設定により任意の機能に変更する事が出来ます。
また、環境変数の種類は次の種類に分類されます。

1. プログラム実行時の機能変更。
2. プログラム実行時の状態表示。
3. 画面編集時の機能変更。

1. プログラム実行時の機能変更

これらの環境変数は、ライブラリの基本動作を変更します。

1.1 Jserver の指定（Wnn）

ネットワーク上で動作している、Jserver を利用する事が可能になります。
環境変数がない場合には localhost にアクセスします。

setenv　JSERVER　ホスト名

1.2 Wnn 初期化処理の指定

通常、Wnnの初期化（Jserverと接続）は、日本語入力開始時に行なっています。
この環境変数の設定を行なう事により、Tkinit関数でJserverとの接続を行ないます。

setenv　KTWNNINI　on

1.3 XIM の利用（XIM 利用可能機種のみ）

日本語入力機能『XIM』の利用が可能となります。
この環境変数が設定されていない場合には、X-Mate デフォルトの日本語入力環境（Wnn）となります。
環境変数設定の際、御使用の日本語入力環境（XIM）のバージョンにより設定値が異なりますので御注意下さい。

《バージョン 9.30以前の場合》 setenv　KTFEP　xim
《バージョン 9.40以降の場合》 setenv　KTFEP　xim2　← 通常はこちらでOKです。

【注意】 確認済みの FEP は、VJE, ATOK です。

また、VJE の利用に限り、次の環境変数を設定で、入力形態の変更が可能となります。

【オーバー ザ スポット】　setenv　XIMSTYLE　position
【ルートウィンドウ】　setenv　XIMSTYLE　root
デフォルトはこの形態です。
他のFEP利用時はこの形態です。

1.4 入力後コールバック制御

標準では入力中にマウス操作や関数操作等で入力フィールドの切り換えが行なわれた際には、後コールバックが呼ばれません。次の環境変数の設定で呼び出す事が可能となります。

setenv　KTISUB　on

1.5 再描画時の強制描画更新

帳票/データ入力部品等では、リフレッシュ等による再描画時の描画負担を軽減する為、部品情報の変更がされていても描画更新（TKdisplay等）を行なっていない限り画面への反映を行なっていません。次の環境変数を設定する事により、再描画時に強制的に画面への描画更新を行ないます。

setenv　KTTBREDSP　on

1.6 ビジュアル属性の選択

起動時に使用するカラーマップのビジュアル属性を選択出来ます。
通常は、デフォルトのカラーマップを使用しています。

setenv	KTVISUAL	default	:	デフォルトのカラーマップを使用
		PseudoColor	:	スードカラーマップを使用
		StaticColor	:	スタティックカラーマップを使用
		DirectColor	:	ダイレクトカラーマップを使用
		TrueColor	:	トゥルーカラーマップを使用
		GrayScale	:	グレイスケールカラーマップを使用
		StaticGray	:	スタティックグレイカラーマップを使用

1.7 スクロールバーを Motif ライクにする

次の環境変数の設定により、スクロールバーの形状及び操作性が Motif と同様の動作を行ないます。

setenv KTMOTIF on

1.8 スクロールバーのデフォルト幅変更機能（自動付属の部品）

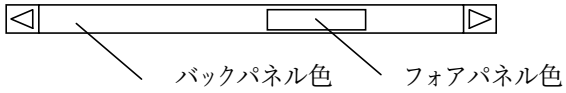
スクロールバーが自動で付加される部品では、スクロールバーの幅は固定サイズでした。これを任意のサイズに指定する事が可能です。
設定範囲は、16〜50 まで。デフォルトは 20

setenv KTSBWIDTH サイズ

1.9 スクロールバー色変更機能（自動付属の部品）

スクロールバーが自動で付加される部品では、スクロールバーの色は部品の色を利用していました。次の環境変数を設定する事で任意の色で表示する事が可能となります。また、全てのスクロールバーが同じ色で統一されます。

setenv KTSBFORE フォアパネル色
setenv KTSBBACK バックパネル色



1.10 スクロールバーのAUTOリピート機能

マウスの左右ボタンを押し続けた際に、連続してページスクロールを行ないます。
Motif ライクの場合には、アローボタン（矢印）を押し続けると連続してスクロールします。

setenv KTSBREP on

1.11 新帳票部品スクロール描画方式の変更機能

新帳票部品のスクロール操作を行なった際に、帳票内を消去してから再描画を行なっています。この為、表示がちらついて見える場合があります。これを解消する方法として2通りの方式を用意しました。

setenv KTTBNOCL モード
- 1： 消去のみ行なわない。
- 2： 帳票のイメージを Pixmap パネルにする。

1.12 日本語入力時の機能キーを変更する

機能キーの変更情報を記述したファイルのパスをこの環境変数に設定します。
ファイル記述の詳細は『インストールガイド』X-Mate 動作時 Q&A に記載。

setenv KTKEYMAP ファイルパス

1.13 デフォルトのフォントを変更する

デフォルトのフォント変更情報を記述したファイルのパスをこの環境変数に設定します。
ファイル記述の詳細は『インストールガイド』X-Mate 動作時 Q&A に記載。

setenv KTFONT ファイルパス

2. プログラム実行時の情報表示

これらの環境変数は、X-Mate ライブラリの動作情報を出力します。

2.1 使用しているフォント名称を知る方法

X-Mate が使用しているフォントの名称を知る事が出来ます。
この環境変数を設定してプログラムを実行して下さい。

setenv FONTNAME on

3.　画面編集時の機能変更

これらの環境変数は、X-Mate のエディタを使用する際に関係する環境変数です。

3.1　X-Mate インストールパス設定

エディタを使用する際には、必ず設定しておく必要があります。

setenv　XMATEHOME　X-Mate インストールパス

3.2　rgb.txt ファイル設定

色設定を行なう際に、X が提供しているカラーデータベースファイル（rgb.txt）を参照していますので、このファイルのパスを指定します。この環境変数がない場合には、/usr/lib/X11/rgb.txt を参照します。

setenv　XCOLOR　ファイルパス

3.3　帳票部品の最大行列数の変更

各帳票部品毎に最大行列数を変更します。設定値は行列数共用です。

帳票部品　（KOTABLE）：setenv　XMATE_TABLE　最大数
新帳票部品（KOTABLE2）：setenv　XMATE_TABLE2　最大数

3.4　ファイル名文字数の制限変更

御利用のファイルシステムにファイル名の長さ制限がある場合は、この設定を行なう事で、画面編集により出力されるファイル文字数のチェックを行ないます。
設定出来る値は、14～256 です。この環境変数がない場合には 256 です。

setenv　MATE_FSIZE　文字長

初版発行　　1991 年　　1 月
第八版　　2010 年　　4 月