

pl ライブラリ リファレンスマニュアル

株式会社ループドピクチャー

<http://www.loopedpicture.com>

Copyright 1996-2002 Looped Picture Company. All Rights Reserved.

はじめに

本マニュアルは、株式会社ループドピクチャーで開発されたマイクロソフトウィンドウズ上でOpenGLを使用したグラフィックス・プログラミングを行うための開発支援ライブラリのリファレンス・マニュアルです。

plライブラリを使用してOpenGLのプログラミングを行う方全員が無償で参照することができますが、本マニュアルの使用に関して下記の内容にしたがってください。

1. 「plライブラリ リファレンス マニュアル」(以下マニュアル)の変更、追加を無断で行うことは一切禁止です。
2. 2次配布は自由ですが、2次配布を行った際に生じたトラブル等、一切の責任を株式会社ループドピクチャーは負いません。
3. 本マニュアルを参照したことによって、生じた障害、トラブルに関して一切の責任を株式会社ループドピクチャーは負いません。
4. このマニュアルの商用使用、利益を得る行為すべてを一切禁止といたします。
5. このマニュアルに記載されている、メーカー名、パテント名称は、それらを所有する組織、会社、団体が所有しています。
6. このマニュアルは予告無く、内容を改善することがあります。
7. このマニュアルの著作権は株式会社ループドピクチャーが所有しています。

上記をご理解頂けない場合、このマニュアル、plライブラリに関するすべてのマテリアルを破棄することを義務つけます。

本マニュアルの使い方

plライブラリを使う上で、plライブラリで供給される各関数群の機能を1ページごとに解説しております。plライブラリ付属のサンプルコードと一緒にご覧ください。

void plSethinstance(HINSTANCE hInstance)

関数の説明：

pl ライブラリを使用する際、ウィンドウズアプリケーションのHINSTANCEを登録します。この関数は、pl ライブラリを使う上で一番初めに使用する必要があります。

例：

ウィンドウズのWIN32API を用いたプログラム開発を行う場合、プログラムの一番はじめは、WinMain関数がプログラムの入り口となり、下記のように記載します。pl ライブラリを使用する場合、WinMain 関数の一番はじめに下記のようにplSethinstance 関数を用いて、ウィンドウズOS から渡されるhInstance をpl ライブラリに登録します。

```
int WINAPI WinMain (HINSTANCE hInstance,HINSTANCE hPrevInstance,LPSTR lpszCmdLine, int nCmdShow)
{
    // pl ライブラリへアプリケーションのHINSTANCE の登録
    plSethinstance(hInstance);
    . . .
}
```

plSethinstance を用いてHINSTANCE の指定を行わない場合、pl ライブラリが正常に動作しません。必ずアプリケーションの初めに指定してください。

戻り値：

戻り値はありません。

参照関数：

plGetwindow,plCreatewindow

int plGetwindow(void)

関数の説明：

plライブラリを用いてOpenGL描画用ウィンドウを作成する際にplライブラリが管理するウィンドウ ID を取得します。

複数のウィンドウを使用する場合は、複数回plGetwindow関数を呼び作成するウィンドウ ID を取得します。

例：

```
int WINAPI WinMain (HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpszCmdLine, int nCmdShow)
{
    // pl ライブラリへアプリケーションの HINSTANCE の登録
    plSethinstance(hInstance);

    // pl ライブラリのウィンドウ作成の準備
    // 作成するウィンドウ ID を取得する。
    wid = plGetwindow();
    . . .
}
```

戻り値：

plライブラリでしようできるウィンドウ ID を int 型で返します。ウィンドウ ID はゼロ以上が正常で、作成に失敗した場合、マイナスの値を返します。

参照関数：

plCreatewindow, plSethinstance

HWND plCreatewindow(int plwinid,int x,int y,int width,int height,char *titlename)

関数の説明：

pl ライブラリを用いて OpenGL 描画用ウィンドウを作成します。

引数plwinidは、plGetwindow で取得したウィンドウ IDを指定し、ウィンドウの表示位置をx,yで指定(左上座標が原点) OpenGLで描画する描画エリアの幅と高さをwidth,heightで指定します。

ウィンドウにタイトルバーがある場合、引数titlenameで指定した内容がタイトルバーに表示されます。

例：

```
plCreatewindow(wid,PL_OSMATTER,PL_OSMATTER,200,200,"sample00")
```

上記の例は、plGetwindow関数で取得したwidのウィンドウをタイトル名 sample00、幅・高さ200ピクセルのOpenGLウィンドウを作成します。OpenGLウィンドウは、RGBモード、ダブルバッファモードで作成されます。(将来的にOpenGL描画領域の詳細設定の拡張予定。)

上記の例の引数x,yに指定されている、PL_OSMATTERは、ウィンドウズOSから指定があった位置にウィンドウを作成します。ユーザーが任意の座標を指定した場合その位置にウィンドウを作成します。

この関数はウィンドウを作成しますが、実際に表示はされません。plCreatewindow関数を用いてウィンドウを作成後、plFlush関数を用いて初めて表示・マップされます。

戻り値：

ウィンドウズOSの管理できるHWND形式のウィンドウハンドラを返します。plライブラリを使用する限りこのウィンドウハンドラを参照する必要はありません。plGetwindowで取得されたウィンドウIDのみを管理することでアプリケーション開発が行えます。

参照関数：

plGetwindow,plSetinstance

`void plExitwindow(int plwinid)`

関数の説明：

引数で指定されたウィンドウ ID を持つウィンドウの使用を終了します。

戻り値：

戻り値はありません。

参照関数：

`plGetwindow`, `plCreatewindow`

void plFlush(int plwinid)

関数の説明：

plCreatewindow で作成されたウィンドウを表示、マップします。引数 winid には、plCreatewindow の第 1 引数で指定したウィンドウ ID を指定します。

戻り値：

戻り値はありません。

参照関数：

plCreatewindow

`void plUpdate(int plwinid)`

関数の説明：

引数plwinidで指定されたウィンドウ IDを持つウィンドウの描画更新を行います。イベント駆動型アプリケーションでOpenGLの描画更新を行う際に使用します。

戻り値：

戻り値はありません。

参照関数：

plCreatewindow, plEventLoop

void plEventLoop(int plwinid)

関数の説明：

引数plwinidで指定されたウィンドウの各種イベント処理を行います。
plライブラリを用いてウィンドウを作成した場合アプリケーションのメインループでこの関数を繰り返し呼び出す必要があります。

例：

plライブラリにて1つのウィンドウ作成し、イベント処理を行う場合のメインループ下記の通りです。

```
// メインループ
while(!plGetwinexit(wid))
{
    // plライブラリ標準のイベント処理関数の呼び出し
    plEventLoop(wid);

    // ループでの描画更新
    plUpdate(wid);
}
```

上記の例では、イベントループを1回処理するごとに、OpenGLの描画をplUpdate関数にて強制的に更新しています。plGetwinexit関数がゼロ以外の値を返すまで繰り返し処理されます。

戻り値：

戻り値はありません。

参照関数：

plUpdate, plGetwinexit

```
int plGetwinexit(int plwinid)
```

関数の説明：

引数plwinidで指定されたウィンドウIDを持つウィンドウが終了されたか調べ、ウィンドウに終了通知があった場合、ゼロ以外の値を返します。

例：

```
// メインループ
while(!plGetwinexit(wid))
{
    // pl ライブラリ標準のイベント処理関数の呼び出し
    plEventLoop(wid);

    // ループでの描画更新
    plUpdate(wid);
}
```

上記の例では、plライブラリにてウィンドウイベント処理を行うループの行頭で、ウィンドウの終了状態を確認しています。ゼロ以外の値がplGetwinexitより戻された場合、ループを終了します。

plEventLoop関数を使用して指定されたウィンドウIDを持つウィンドウに終了通知された場合、ゼロ以外の値が返されます。(クローズボックスがクリックされた場合等)

戻り値：

ウィンドウが終了の場合ゼロ以外、終了で無い場合ゼロを返します。

参照関数：

plSetwinexit

`void plSetwinexit(int plwinid)`

関数の説明：

引数で指定したウィンドウ ID を持つウィンドウに終了通知を行います。この関数はユーザーがメニューから「終了」を選択した場合、何かキーを入力した場合にこの関数を使用し、アプリケーションのメインループに終了通知を知らせます。

この関数は、ユーザー作成したアプリケーションが、任意の終了通知を行う必要がある場合に使用します。

戻り値：

戻り値はありません。

参照関数：

`plGetwinexit`

void plSetwinproc(int plwinid,WNDPROC proc)

関数の説明：

plライブラリを用いてウィンドウを作成したとき、作成したウィンドウに対する各種イベント処理を行う関数を指定します。

この関数を用いてウィンドウ処理を行う場合、ユーザーが自分自身でウィンドウイベント処理関数を作成する必要があります。

引数plwinidには、plCreatewindowで作成したウィンドウIDを指定し、WNDPROCには、ウィンドウイベントを処理する関数ポインタを指定します。

通常、この関数を使用してウィンドウイベント処理を行う場合は、高度なアプリケーション開発を行うときに限りられます。plライブラリは標準のウィンドウイベント処理を行う関数が組み込まれていますので、WIN32APIを熟知されていない方はこの関数の使用は避けてください。

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。他の方法でウィンドウを作成した場合、この関数で指定された関数ポインタは無効です。

戻り値：

戻り値はありません。

参照関数：

plCreatewindow

void plSetdrawfunc(int plwinid,PROC drawfunc)

関数の説明：

plライブラリで作成したウィンドウのOpenGL描画を行うための描画更新関数を定義します。

引数plwinidには、plGetwindowで取得したウィンドウ ID、引数drawfuncには、ユーザーが用意した描画関数ポインタを指定します。

例：

```
PL_FUNC drawfunc(void)
{
    // OpenGL の描画
}

{
    . . .
    // OpenGL の描画関数の指定
    plSetdrawfunc(wid,&drawfunc);
    . . .
}
```

上記の例では、widで指定されたウィンドウ IDを持つウィンドウのOpenGL描画関数として、drawfuncを指定しています。描画関数はPL_FUNC型で定義しなければなりません。

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetinitfunc,plCreatewindow

void plSetinitfunc(int plwinid,PROC initfunc)

関数の説明：

plライブラリで作成したウィンドウのOpenGL初期化を行うための初期化関数を定義します。

引数plwinidには、plGetwindowで取得したウィンドウ ID、引数 initfuncには、ユーザーが用意した描画関数ポインタを指定します。

例：

```
PL_FUNC initfunc(void)
{
    // OpenGL の初期化
}

{
    . . .
    // OpenGL の初期化関数の指定
    plSetinitfunc(wid,&initfunc);
    . . .
}
```

上記の例では、widで指定されたウィンドウ IDを持つウィンドウのOpenGL初期化関数として、initfuncを指定しています。初期化関数はPL_FUNC型で定義しなければなりません。この関数は、ウィンドウのリサイズ、描画更新イベントを受信した際、plライブラリ標準のイベント処理関数から呼び出されます。

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetdrawfunc,plCreatewindow

void plSetcommandfunc(int plwinid,PROC commandfunc)

関数の説明：

引数 plwinid で指定されるウィンドウ ID を持つウィンドウのコマンド処理関数を commandfunc にて指定します。

この関数は、pl ライブラリにて、メニューを持つウィンドウを作成した場合に定義し、commandfunc で指定された関数の中では、選択されたメニューアイテムに対して処理を行う必要があります。

例：

```
PL_FUNC CommandProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    // メニューアイテムが選択されたときの処理関数
    switch (LOWORD(wParam))
    {
        case ID_MENUITEM40001:
            // ID_MENUITEM40001 は resource.h に定義されている。
            // 終了メニューが選択されたとき。
            plSetwinexit(wid);
            break;
    }
    return 0;
}
```

上記は、メニューアイテム ID、ID_MENUITEM40001 を持つメニューがユーザによって選択された場合ウィンドウの終了を通知するという内容になっています。

コマンド処理関数は PL_FUNC 型で定義しなければなりません。この関数は、plCreatewindow にてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetdrawfunc,plSetinitfunc,plCreatewindow

void plSetkeydownfunc(int plwinid,PROC keydownfunc)

関数の説明：

引数plwinidで指定されるウィンドウIDを持つウィンドウのキーボード入力によるイベント処理を行う関数を keydownfunc に指定します。

キーボード入力処理(メニューなどのショートカットを除く)する必要のあるアプリケーションのみで使用します。

例：

```
PL_FUNC KeydownProc(int keydata)
{
    switch(keydata)
    {
        case PL_KEY_ESC:
            . . .
```

上記の例は、ウィンドウ内部でESCキーが押されたときの処理となります。キー入力に関する値は、plライブラリのインクルードファイルpl.hを参照してください。

上記のKeydownProc関数の引数keydataは入力されたキーの値が収容されこの関数が呼び出されます。

キーボード入力処理関数はPL_FUNC型で定義しなければなりません。この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetdrawfunc,plSetinitfunc,plSetcommandfunc,plCreatewindow

void plSetmousedownfunc(int plwinid,PROC mousedownfunc)

関数の説明：

引数plwinidで指定されるウィンドウIDを持つウィンドウ内部でマウスを押した状態のイベント処理を行う関数をmousedownfuncに指定します。

マウス入力処理入力処理（メニューメニューを選ぶなどを除く）する必要のあるアプリケーションのみで使用します。

例：

```
PL_FUNC MousedownProc(int btype,int x,int y)
{
    . . .
}
```

上記のMousedownProc関数の引数btypeには、マウスが押されたときの値、x,yにはマウスが押された座標が収容されこの関数が呼び出されます。引数btypeの内容は、pl.h内部で定義されている下記の4つを判定することで処理します。マウスボタンの判定には、plDrawend関数を使用します。

PL_MOUSE_LEFT	左ボタン
PL_MOUSE_RIGHT	右ボタン
PL_MOUSE_MIDDLE	中央ボタン
PL_MOUSE_ZMOUSE	マウスホイールを動かしたとき

マウスダウン操作処理関数はPL_FUNC型で定義しなければなりません。この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetdrawfunc,plSetinitfunc,plSetcommandfunc,plSetkeydownfunc,plSetmouseupfunc,plDrawend,plCreatewindow

```
void plSetmouseupfunc(int plwinid,PROC mouseup_proc)
```

関数の説明：

引数plwinidで指定されるウィンドウIDを持つウィンドウ内部でマウスボタンを押した後、ボタンを離れた状態のイベント処理を行う関数を mousedownfunc に指定します。

マウス入力処理入力処理（メニューメニューを選ぶなどを除く）する必要のあるアプリケーションのみで使用します。

例：

```
PL_FUNC MouseupProc(int btype,int x,int y)
{
    . . .
}
```

上記の MouseupProc 関数の引数 btype には、マウスが押されたときの値、x,y にはマウスが押された座標が収容されこの関数が呼び出されます。引数 btype の内容は、pl.h 内部で定義されている下記の 4 つを判定することで処理します。マウスボタンの判定には、plDrawend 関数を使用します。

PL_MOUSE_LEFT	左ボタン
PL_MOUSE_RIGHT	右ボタン
PL_MOUSE_MIDDLE	中央ボタン
PL_MOUSE_ZMOUSE	マウスホイールを動かしたとき

マウスアップ操作処理関数は PL_FUNC 型で定義しなければなりません。この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetdrawfunc,plSetinitfunc,plSetcommandfunc,plSetkeydownfunc,plSetmousedownfunc,plDrawend,plCreatewindow

`void plSwapbuffer(void)`

関数の説明：

plライブラリで作成されたOpenGL描画領域をスワップバッファします。plライブラリで作成されるOpenGL描画ウィンドウは、ダブルバッファモードで作成されるため、描画関数の中でフロントバッファとバックバッファを入れ替える必要があります。

ダブルバッファに関する詳細は、OpenGL Programing Guide(赤本)を参照してください。

戻り値：

戻り値はありません。

参照関数：

plDrawstrat, plDrawend, plCreatewindow

`int piGetwindowid(HWND hWnd)`

関数の説明：

引数 `hWnd` で指定される WIN32API のウィンドウハンドラから `pi` ライブラリのウィンドウ ID を取得します。この引数で指定できるウィンドウハンドラは `piCreatewindow` を使用して作成したウィンドウに限られます。

`piCreatewindow` でウィンドウ作成時に、WIN32API の `HWND` 型のウィンドウハンドルを管理している場合にこの関数を使用し、ウィンドウ ID を調べる際に使用します。

戻り値：

関数が成功すると、ウィンドウ ID が返されます。`pi` ライブラリ以外で作成されたウィンドウの場合、負の値が返されます。

参照関数：

`piGetwindow`, `piCreatewindow`, `piGetwindowhwnd`

HWND plGetwindowhwnd(int plwinid)

関数の説明：

引数で指定されたウィンドウ ID の HWND 型のウィンドウハンドラを取得します。

戻り値：

WIN32API で定義されている HWND 型のウィンドウハンドラ。失敗し場合は NULL を返します。

参照関数：

plGetwindow, plCreatewindow, plGetwindowid

```
void plGetwindowSize(int plwinid,int *width,int *height)
```

関数の説明：

引数plwinidで指定されたウィンドウのウィンドウ位置、サイズをwidth,heightに取得します。

戻り値：

戻り値はありません。

参照関数：

void plSetnoframe(int plwinid)

関数の説明：

plCreatewindow 関数にてウィンドウを作成するときに、作成するウィンドウをタイトル無し、フレーム無しで作成するように指定します。引数plwinidにはplGetwindow関数で取得したウィンドウ IDを指定します。

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plGetwindow,plCreatewindow,plGetnoframe

`int plGetnoframe(int plwinid)`

関数の説明：

引数plwinidで指定されたウィンドウIDを持つウィンドウがタイトル無しおよびフレーム無しの場合1、そうでなければゼロを返します。

戻り値：

1の場合、引数で指定されたウィンドウはノーフレーム、ノータイトルウィンドウの指定がされています。その他の場合はゼロです。

参照関数：

plGetwindow,plCreatewindow,plSetnoframe

void plSetfullscreen(int plwinid)

関数の説明：

引数plwinidで指定されたウィンドウをフルスクリーンサイズに設定します。フルスクリーンモードの場合、タイトルバー、ウィンドウフレームは表示されません。

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plGetfullscreen

`int plGetfullscreen(int plwinid)`

関数の説明：

引数plwinidで指定されたウィンドウIDを持つウィンドウがフルスクリーン指定の場合1、そうでなければゼロを返します。

戻り値：

フルスクリーンモードの場合は1、そうでなければゼロを返します。

参照関数：

plSetfullscreen

```
int plDisplaymode(int width,int height,int depth, int hz)
```

関数の説明：

ディスプレイの表示モードを引数で指定した内容に基づいて変更します。この関数はフルスクリーンモードでの描画時、ディスプレイの表示モードを変更するときに使用します。

引数で指定されたウィンドウサイズ、周波数への変更に失敗すると1、成功するとゼロを返します。

例：

```
plDisplaymode(640,480,24,60);
```

この例では、640x480のVGAサイズ60Hz、24ビットカラーモードへ変更します。

指定されたレゾリューション、周波数での表示が行えるグラフィックス能力を持つビデオカードを使用していない場合、何も処理されません。

この関数はアプリケーションで1回目の使用のとき、引数で指定されるレゾリューションに変更し、2回目の使用以降は、元のサイズ(この関数が使用される前の状態)に戻します。

この関数は、plGetfullscreenにてフルスクリーンサイズのウィンドウを作成後、ディスプレイモードの変更を行う場合に使用します。ウィンドウズ上でOpenGLを使用し、低いレゾリューションで描画を行う場合(ゲームアプリケーション等)の開発に適しています。

戻り値：

成功の場合ゼロ、失敗の場合1を返します。

参照関数：

plGetfullscreen

void plSeticon(int plwinid,int iconid)

関数の説明：

引数plwinidで指定されたウィンドウ IDのアイコン ID(32x32)を指定します。

アイコンはVisua Cのリソースエディタで作成したリソース番号を指定します。(Visual Cで作成されるresource.hをインクルードすること。リソースエディタを使用して編集した場合、resource.hが自動的に作成されます。)

ここで指定されたアイコンが、デスクトップ上で表示される大きなサイズのアイコンとして表示されます。

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetSmallIcon

void plSetSmallIcon(int plwinid,int siconid)

関数の説明：

引数plwinidで指定されたウィンドウ IDのsmallサイズアイコン ID(16x16 ピクセル)を指定します。

アイコンはVisual Cのリソースエディタで作成したリソース番号を指定します。(Visual Cで作成されるresource.hをインクルードすること。リソースエディタを使用して編集した場合、resource.hが自動的に作成されます。)

ここで指定されたアイコンが、デスクトップ上で表示される小さなサイズのアイコンとして表示されます。

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plSetIcon

```
void plSetmenu(int plwinid,unsigned short menuid)
```

関数の説明：

引数plwinidで指定されるウィンドウにメニューを追加するとき、引数menuidに指定されたメニューを追加します。

メニュー IDは、Visual Cのリソースエディタで作成されたIDを指定します。(Visual Cで作成されるresource.hをインクルードすること。リソースエディタを使用して編集した場合、resource.hが自動的に作成されます。)

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

plGetmenu

`unsigned short plGetmenu(int plwinid)`

関数の説明：

引数plwinidで指定されたウィンドウIDを持つウィンドウに指定されたメニューIDを取得します。

ウィンドウへメニューの追加は、plSetmenu関数にて行います。

戻り値：

メニューIDを返します。取得に失敗、メニューがついていないウィンドウへこの関数を実行すると-1を返します。

参照関数：

plSetmenu

void plSetMenuItem(int plwinid,int menuid,int mtype)

関数の説明：

引数plwinidで指定されたウィンドウ IDを持つウィンドウのメニューアイテムの状態を設定します。引数menuidには、VisualCのリソースエディタで作成したリソース IDを指定します。

引数mtypeには、pl.hで定義されている下記の3つが指定できます。

PL_MENU_ENABLE	メニューアイテムを使用出来るように指定
PL_MENU_DISABLE	メニューアイテムを使用出来ないように指定
PL_MENU_GRAY	メニューアイテムを使用出来ないようにグレー表示指定

戻り値：

戻り値はありません。

参照関数：

plSetMenucheck

```
void plSetMenucheck(int plwinid,int menuid,int check)
```

関数の説明：

引数plwinidで指定されたウィンドウ IDを持つウィンドウのメニューアイテムに対してチェックのオン・オフ指定を行います。引数menuidには、VisualCのリソースエディタで作成したリソース IDを指定します。

引数 check には、pl.h で定義されている下記の2つが指定できます。

PL_MENU_CHECK	メニューにチェックを入れます。
PL_MENU_UNCHECK	メニューのチェックを外します。

戻り値：

戻り値はありません。

参照関数：

plSetMenuItem

void plSetAccelerators(int plwinid,unsigned short accelerator)

関数の説明：

引数plwinidで指定されたウィンドウIDを持つメニューのショートカットを定義するアクセラレーターIDを引数acceleratorで指定します。

アクセラレーターIDは、Visual Cのリソースエディタで作成されたIDを指定します。(Visual Cで作成されるresource.hをインクルードすること。リソースエディタを使用して編集した場合、resource.hが自動的に作成されます。)

この関数は、plCreatewindowにてウィンドウを作成する前に使用することが限定されています。

戻り値：

戻り値はありません。

参照関数：

`void plDrawstrat(int plwinid)`

関数の説明：

引数 `plwinid` で指定されるウィンドウで OpenGL の描画を行う場合この関数を使用する必要があります。この関数の後、OpenGL の各関数が使用することができます。

この関数は、`plSetinitfunc`、`plSetdrawfunc` で指定される OpenGL の初期化、描画関数の中では使用する必要はありません。その他の関数の中でユーザーが任意で描画する場合には使用します。

`plDrawstrat` を使用したら、必ず `plDrawend` 関数を使用して OpenGL の描画の終了通知を行う必要があります。

`plDrawstrat` を指定後、`plDrawend` を使用して OpenGL 描画終了を行わない場合、動作不良を起こす場合があります。描画の終了が終わったら `plDrawend` を使用して OpenGL の描画終了通知を行ってください。

戻り値：

戻り値はありません。

参照関数：

`plDrawend`, `plSwapbuffer`

void plDrawend(int plwinid)

関数の説明：

引数plwinidで指定されるウィンドウでOpenGLの描画終了通知を行います。OpenGLの描画開始は、plDrawstrat を使用します。

詳細は、plDrawstrat を参照してください。

戻り値：

戻り値はありません。

参照関数：

plDrawstrat, plSwapbuffer

int pIMousebutton(int mbutton, int checkbutton)

関数の説明：

pISetmouseupfunc, pISetmousedownfunc関数で定義されたマウスボタンイベント処理関数の中で引数に指定されたマウスボタンの状態を調べます。

引数 checkbutton には、下記の値を指定することができます。

PL_MOUSE_LEFT	左ボタン
PL_MOUSE_RIGHT	右ボタン
PL_MOUSE_MIDDLE	中央ボタン
PL_MOUSE_ZMOUSE	マウスホイールを動かしたとき

例：

```
PL_FUNC MousedownProc(int btype, int x, int y)
{
    // マウスボタン個別処理を行う場合、下記のように押されているボタンによって判定します。
    if (pIMousebutton(btype, PL_MOUSE_LEFT))
    {
        . . .
    }
    . . .
}
```

上記の例は、マウスが押されたときに処理される MousedownProc関数内部で引数btypeの値を pIMousebutton 関数で判別しています。

戻り値：

checkbuttonに指定されているマウスボタンが有効の場合 1、そうでなければゼロを返します。

参照関数：

pISetmouseupfunc, pISetmousedownfunc

```
void plGetMousedownpos(int plwinid,int checkbutton,int *mx,int *my)
```

関数の説明：

引数plwinidを持つウィンドウのOpenGL描画領域で一番最初にマウスボタンが押された座標を引数mx,myに取得します。

引数 checkbutton には、下記の内容を指定することができます。

PL_MOUSE_LEFT	左ボタン
PL_MOUSE_RIGHT	右ボタン
PL_MOUSE_MIDDLE	中央ボタン

この関数は、plSetmouseupfunc,plSetmousedownfunc関数などで定義されたマウス処理関数の中で、マウスの移動量の計算を行う際に使用します。

この関数はplSetwinprocなどでユーザーが独自のウィンドウ処理関数を定義した場合は機能しません。

戻り値：

戻り値はありません。

参照関数：

plSetmouseupfunc,plSetmousedownfunc,plMouseButton

void plPyramid(float scaling)

関数の説明：

OpenGL描画領域に四角錐を描画します。描画する四角錐のサイズを引数scalingにて指定します。

戻り値：

戻り値はありません。

参照関数：

plSphere,plCube,plGridXZ,plGridXY,plGridZY,plOrigin

```
void plSphere(float scaleing,int div)
```

関数の説明：

OpenGL描画領域に球体を描画します。描画する球体のサイズを引数scaleingにて指定し、球体表面の分割数をdivに指定します。

戻り値：

戻り値はありません。

参照関数：

plPyramid,plCube,plGridXZ,plGridXY,plGridZY,plOrigin

void plCube(float scaleing)

関数の説明：

OpenGL描画領域に正立方体を描画します。描画する正立方体のサイズを引数scaleingにて指定します。

戻り値：

戻り値はありません。

参照関数：

plSphere,plPyramid,plGridXZ,plGridXY,plGridZY,plOrigin

void plGridXZ(float scaleing)

関数の説明：

OpenGL 描画領域の XZ 軸に対してグリッドを描画します。描画するグリッドの1辺のサイズを引数 scaleing にて指定します。

戻り値：

戻り値はありません。

参照関数：

plSphere, plPyramid, plCube, , plGridXY, plGridZY, plOrigin

void plGridXY(float scaleing)

関数の説明：

OpenGL 描画領域の XY 軸に対してグリッドを描画します。描画するグリッドの1辺のサイズを引数 `scaleing` にて指定します。

戻り値：

戻り値はありません。

参照関数：

`plSphere`, `plPyramid`, `plCube`, `plGridXZ`, `plGridZY`, `plOrigin`

void plGridZY(float scaleing)

関数の説明：

OpenGL 描画領域の ZY 軸に対してグリッドを描画します。描画するグリッドの1辺のサイズを引数 scaleing にて指定します。

戻り値：

戻り値はありません。

参照関数：

plSphere, plPyramid, plCube, , plGridXZ, plGridXY, plOrigin

void plOrigin(float scaleing)

関数の説明：

OpenGL 描画領域に原点に対する XYZ 軸を描画します。描画する軸の長さを引数 scaleing にて指定します。

戻り値：

戻り値はありません。

参照関数：

plSphere, plPyramid, plCube, plGridXZ, plGridXY, plGridZY

void plSetfilebutton(char *bstr)

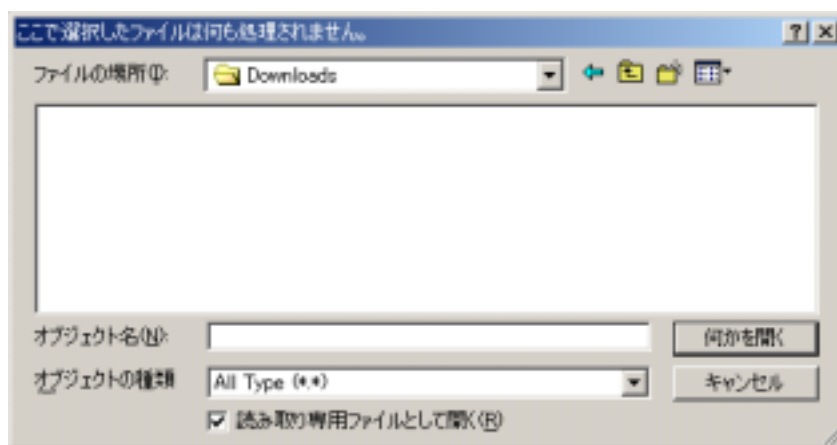
関数の説明：

plGetFile関数を使用してファイルセレクトダイアログを使用するとき、「OK」ボタンのボタンに表示される文字列を設定します。

例：

```
plSetfilebutton("何かを開く");
```

上記の指定の後、plGetFileを使用すると、下記のダイアログのようにボタンの名前が変更されます。



戻り値：

戻り値はありません。

参照関数：

plGetFile, plSetFileType, plAddFileType

```
void plSetFileType(char *fileypestr,char *wildcard)
```

関数の説明：

plGetFile関数を使用してファイルセレクトダイアログを使用するとき、選択するファイルのファイル・タイプを指定します。引数fileypestrには、ダイアログに表示される文字列、引数wildcardには、ファイルの一覧に表示したいワイルドカード文字列を指定します。

例：

```
plSetFileType("All Type (*.*)","*.");
```

上記の例では、ダイアログのファイルタイプとして、All Type (*.*)を表示し、選択されているフォルダ内部のすべてのファイルタイプを表示します。例えば、拡張子 ".jpg" というファイルタイプを選択する場合、引数wildcardに "*.jpg" と指定することによって、拡張子 ".jpg" を持つファイルの一覧が表示されます。

戻り値：

戻り値はありません。

参照関数：

plSetfilebutton,plAddFileType,plGetFile

```
void plAddFileType(char *fileypestr,char *wildcard)
```

関数の説明：

plGetFile関数を使用してファイルセレクトダイアログを使用するとき、選択するファイルのファイル・タイプを追加します。plSetFileTypeにて指定したファイルタイプ以外に、複数のファイルタイプを選択できるようにする際、plAddFileType関数を使用して選択したいファイルのワイルドカードを指定します。

例：

```
plAddFileType("JPEG (*.jpg)","*.jpg");
```

上記の例では、ダイアログのファイルタイプ拡張子 "*.jpg" のファイル一覧を選択できるように追加しています。

戻り値：

戻り値はありません。

参照関数：

plSetfilebutton,plSetFileType,plGetFile

```
int plGetFile(int plwinid,char *title,char *filename,int ftype)
```

関数の説明：

引数 plwinid に対して、ファイルセレクトダイアログを表示します。

引数 title にはファイルセレクトウィンドウに表示されるタイトル名を指定します。

引数 filename には、選択されたファイル名がフルパスで収容されます。(plGetFile が 1 を返した場合) filename の文字列のサイズは、pl.h で定義されている PL_MAXSTR を必ず指定してください。

引数 ftype には、下記の値が指定することができます。

PL_READ_FILE ファイルを読み込む際に指定
PL_WRITE_FILE ファイルを書き込む際に指定

例：

```
char fname[PL_MAXSTR];
plSetFilebutton("開く");
plSetFileType("All Type (*.*)","*.");
plAddFileType("JPEG (*.jpg)","*.jpg");
if (plGetFile(wid,"イメージファイルの選択",fname,PL_READ_FILE) == PL_OK)
{
    // ファイルの指定あり
    . . .
}
```

戻り値：

ファイルの指定が行われた場合 1、キャンセルされた場合ゼロを返します。

参照関数：

plSetFilebutton,plSetFileType,plAddFileType

```
void plNormalv(float *k0,float *k1,float *k2,float *N)
```

関数の説明：

3頂点より面法線を計算します。引数 k0[3],k1[3],k2[3]に3頂点を指定します。計算された法線はN[3]に出力されます。

頂点の並びはX,Y,Zです。

この関数の引数は配列指定です。

戻り値：

戻り値はありません。

参照関数：

plNormal

```
plNormal(float x0,float y0,float z0,float x1,float y1,float z1,float x2,float  
y2,float z2,float *nx,float *ny,float *nz);
```

関数の説明：

3頂点より面法線を計算します。面を構成する3頂点x0,y0,z0,x1,y1,z1,x2,y2,z2, から面法線を計算し、*nx,*ny,*nzに計算結果が出力されます。

頂点を配列指定する場合は、plNormalvを使用してください。

戻り値：

戻り値はありません。

参照関数：

plNormalv

pl ライブラリ関数一覧

ウィンドウ作成関数

```
void plSethinstance(HINSTANCE hInstance)
int plGetwindow(void)
HWND plCreatewindow(int plwinid,int x,int y,int width,int height,char *titlename)
void plExitwindow(int plwinid)
void plFlush(int plwinid)
void plUpdate(int plwinid)
void plEventLoop(int plwinid)
int plGetwinexit(int plwinid)
void plSetwinexit(int plwinid)
```

ウィンドウ処理関数

```
void plSetwinproc(int plwinid,WNDPROC proc)
void plSetdrawfunc(int plwinid,PROC drawfunc)
void plSetinitfunc(int plwinid,PROC initfunc)
void plSetcommandfunc(int plwinid,PROC commandfunc)
void plSetkeydownfunc(int plwinid,PROC keydownfunc)
void plSetmousedownfunc(int plwinid,PROC mousedownfunc)
void plSetmouseupfunc(int plwinid,PROC mouseup_proc)
```

ウィンドウ情報取得関数

```
int plGetwindowid(HWND hWnd)
HWND plGetwindowhwnd(int plwinid)
void plGetwindowsize(int plwinid,int *width,int *height)
```

ウィンドウ設定関数

```
void plSetnoframe(int plwinid)
int plGetnoframe(int plwinid)
void plSetfullscreen(int plwinid)
int plGetfullscreen(int plwinid)
int plDisplaymode(int width,int height,int depth, int hz)
```

メニュー関数

```
void plSetmenu(int plwinid,unsigned short menuid)
unsigned short plGetmenu(int plwinid)
void plSetmenuitem(int plwinid,int menuid,int mtype)
void plSetmenucheck(int plwinid,int menuid,int check)
void plSetAccelerators(int plwinid,unsigned short accelerator)
```

マウス関数

```
int plMouseButton(int mbutton,int checkbutton)
void plGetMouseDownpos(int plwinid,int checkbutton,int *mx,int *my)
```

OpenGL の描画関数

```
void plDrawstrat(int plwinid)
void plDrawend(int plwinid)
void plSwapbuffer(void)
```

OpenGL オブジェクト描画関数

```
void plPyramid(float scaleing)
void plSphere(float scaleing,int div)
void plCube(float scaleing)
void plGridXZ(float scaleing)
void plGridXY(float scaleing)
void plGridZY(float scaleing)
void plOrigin(float scaleing)
```

ファイルセレクトダイアログ関数

```
void plSetfilebutton(char *bstr)
void plSetFileType(char *fileypestr,char *wildcard)
void plAddFileType(char *fileypestr,char *wildcard)
int plGetFile(int plwinid,char *title,char *filename,int ftype)
```

算術関数

```
void plNormalv(float *k0,float *k1,float *k2,float *N);
DLL_API plNormal(float x0,float y0,float z0,float x1,float y1,float z1,float x2,float y2,float
z2,float *nx,float *ny,float *nz);
```

pl ライブラリ情報取得関数一覧

```
HWND plCreatewindow(int plwinid,int x,int y,int widht,int height,char *titlename)
int plGetFile(int plwinid,char *title,char *filename,int ftype)
int plGetfullscreen(int plwinid)
unsigned short plGetmenu(int plwinid)
void plGetMouseDownpos(int plwinid,int checkbutton,int *mx,int *my)
int plGetnoframe(int plwinid)
int plGetwindow(void)
HWND plGetwindowhwnd(int plwinid)
int plGetwindowid(HWND hwnd)
void plGetwindowsize(int plwinid,int *width,int *height)
int plGetwinexit(int plwinid)
int plMouseButton(int mbutton,int checkbutton)
```

pl ライブラリ設定関数一覧

```
void plAddFileType(char *fileypestr,char *wildcard)
int plDisplaymode(int width,int height,int depth, int hz)
void plDrawend(int plwinid)
void plDrawstrat(int plwinid)
void plEventLoop(int plwinid)
void plExitwindow(int plwinid)
void plFlush(int plwinid)
void plSetAccelerators(int plwinid,unsigned short accelerator)
void plSetcommandfunc(int plwinid,PROC commandfunc)
void plSetdrawfunc(int plwinid,PROC drawfunc)
void plSetfilebutton(char *bstr)
void plSetFileType(char *fileypestr,char *wildcard)
void plSetfullscreen(int plwinid)
void plSethinstance(HINSTANCE hInstance)
void plSetinitfunc(int plwinid,PROC initfunc)
void plSetkeydownfunc(int plwinid,PROC keydownfunc)
void plSetmenu(int plwinid,unsigned short menuid)
void plSetmenuItem(int plwinid,int menuid,int mtype)
void plSetmenucheck(int plwinid,int menuid,int check)
void plSetmousedownfunc(int plwinid,PROC mousedownfunc)
void plSetmouseupfunc(int plwinid,PROC mouseup_proc)
void plSetnoframe(int plwinid)
void plSetwinexit(int plwinid)
void plSetwinproc(int plwinid,WNDPROC proc)
void plSwapbuffer(void)
void plUpdate(int plwinid)
```

pl ライブラリ画関数一覧

```
void plPyramid(float scaleing)
void plCube(float scaleing)
void plSphere(float scaleing,int div)
void plCube(float scaleing)
void plGridXZ(float scaleing)
void plGridXY(float scaleing)
void plGridZY(float scaleing)
void plOrigin(float scaleing)
```

pl ライブラリ算術関数

```
void plNormalv(float *k0,float *k1,float *k2,float *N);
DLL_API plNormal(float x0,float y0,float z0,float x1,float y1,float z1,float x2,float y2,float
z2,float *nx,float *ny,float *nz);
```