ソフトゼミβ　第2回　画像処理と移動

今回のβではDXライブラリの新出関数の解説をしていきます。
この他にも沢山関数があるので、詳細が知りたい方はDXライブラリのHP
http://homepage2.nifty.com/natupaji/DxLib/の
「DXライブラリの関数リファレンスマニュアル」から見ることができます。
インターネットが使えない環境ではDXライブラリのフォルダ→「help」→「index.html」からも見ることができます。

■ 新出関数の解説

➢ ゲームループに使った関数

ProcessMessage関数

int ProcessMessage(void) ;

ウインドウが閉じられたときや、エラーが発生した時はプログラムを終了しなければなりません。この関数はwindowsからのそういったメッセージを受け取ります。この重要な処理を行うためゲームループするごとに1回程度の頻度で必ず呼び出す必要があります。

CheckHitKey関数

int CheckHitKey(int KeyCode) ;

キーが押されているかどうかを調べる関数です。引数でどのキーを調べるか指定します。戻り値が1なら押されていて、0なら押されていません。引数のリストはリファレンスにありますのでここでは一部を抜粋します。

KEY_INPUT_ESCAPE //エスケープキー	KEY_INPUT_A //Aキー
KEY_INPUT_B //Bキー	
KEY_INPUT_UP //上キー	以下同様
KEY_INPUT_DOWN //下キー	
KEY_INPUT_RIGHT //右キー	KEY_INPUT_1 //１キー
KEY_INPUT_LEFT //左キー	KEY_INPUT_2 //２キー
以下同様	

今回はProcessMessage関数とCheckHitKey関数を使ってゲームループのループ条件を
while(ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0)
「プログラムが正常に動作していて、かつESCAPEキーが押されていない」にしています。
なお、KEY_INPUT_～はDxLib.h内にある#defineで定義されていて、実際にはint型の値になっています。

- 画像・描画に関係する関数

LoadGraph 関数

int LoadGraph(char *FileName) ;

LoadGraph 関数は引数内の文字列に従って画像を読み込み、識別番号を付けます。
この番号は(グラフィック)ハンドルと言い、int 型の数値です。取り込んだ画像を使うときにはこのハンドルを使います。今回は構造体 imgList のメンバである player にこの数を格納しています。

SetDrawScreen関数

int SetDrawScreen(int DrawScreen) ;

この関数は描画する場所を表画面(そのまま描写)、裏画面(ユーザーからは見えない裏の画面)のどちらにするか設定します。今回は裏画面に描写します。表に描写すると描写途中の画像が見え、ちらついているように感じることがあり、これを防ぐため、裏画面に描写します。裏画面を指定する場合は、SetDrawScreen(DX_SCREEN_BACK);のように指定します。

CleanDrawScreen関数

int ClearDrawScreen(void) ;

各種描画関数で描画したグラフィックをすべて消し画面を初期化します。

DrawGraph関数

int DrawGraph(int x,int y,int handle,int flag) ;

というようになっていて、(x, y)を左上の起点にしてhandle(先ほどのグラフィックハンドル)で指定した画像を描写します。最後の引数は透明色を使うかのフラグでTRUEにすれば透明色が有効になります。

GetColor関数

int GetColor(int Red , int Green , int Blue) ;

RGBの数字を入れることで、DrawString関数などで使う色コードを出力してくれます。出したい色のRGBがわからないという人はペイントツールなどのパレットで調べてください。

DrawString関数

int DrawString(int x , int y , char *String , int Color)

画面内の座標(x,y)を左上としてStringに指定された文字列を描写します。文字の色は上のGetColor関数で指定してください。

DrawFormatString関数

```
int DrawFormatString( int x , int y , int Color ,char *FormatString , ... ) ;
```

DrawStringと同じですが、printfのように%dなどで出力に変数を使えます。DrawString関数と引数の順番が違うので注意してください。

ScreenFlip関数

```
int ScreenFlip( void ) ;
```

裏画面に描かれている画像を表画面に反映します。

GetHitKeyStateAll関数

int GetHitKeyStateAll(char *KeyStateBuf)

特定のキーが押されているかどうかを判定するために、CheckHitKeyという関数を紹介しました。しかし、同じフレーム内で2回以上同じキーの状態を見るときにCheckHitKeyを2回呼び出しているようでは効率が悪いです。

そこで、GetHitKeyStateAll関数の登場です。この関数は、引数にchar型、要素数256個の配列(実際のところはポインタですが、ここでは配列として説明)を指定すると、その配列の各要素に、対応するキーの入力状態(押されていない: 0, 押されている: 1)が代入されます。

例えば、char keyState[256];という宣言がある時に、キーの状態を取得する時には、

```
GetHitKeyStateAll( keyState );
```

と書きます。その中で、Zキーが押されているかどうかを見る時には、

```
if( keyState[ KEY_INPUT_Z] ){
//押されている時の処理
}else{
//押されていない時の処理
}
```

という風に書きます。

※ KEY_INPUT～というのはDxLib.hで定義されている定数で、各キーの入力状態を管理する配列の添字の番号です。例えば、Zキーを管理する添字の番号はKEY_INPUT_Zの値で、10進数で44(16進数で0x2C)です。

※ KEY_INPUT～と実際のキーボードとの対応は CheckHitKey と同じです。CheckHitKey の説明を見てください。

- その他の重要な関数

次の関数はプログラムに出てきませんが、重要なので覚えておくといいかもしれません。

SetFontSize関数

int SetFontSize(int FontSize) ;

DrawStringなどの文字のサイズを変更します。引数は文字のサイズで、おおよそドット単位で指定します。適当な値を入れて自分の好きな大きさに調整してください。

<使用例>

```
#include "DxLib.h"
#define WINDOW_WIDTH 640
#define WINDOW_HEIGHT 480
int WINAPI WinMain( HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd ){
//初期化
ChangeWindowMode( TRUE ) ;
SetGraphMode( WINDOW_WIDTH,WINDOW_HEIGHT, 32 );
if( DxLib_Init() == -1 ){ return -1; }
//ゲームループ
while(ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0){
ClearDrawScreen(); //画面を初期化
SetFontSize(15); //フォントのサイズを変更
DrawString(100,100, "15のサイズで描写",GetColor(255,255,255)); //文字出力
SetFontSize(30); //フォントのサイズを変更
DrawString(100,200, "30のサイズで描写",GetColor(255,255,255)); //文字出力
ScreenFlip(); //裏画面を表画面に反映
}
DxLib_End();
return 0;
}
```

サイズを15にして「15のサイズで描写」と出力する。

サイズを30にして「30のサイズで描写」と出力する。

という処理をしています。