ソフトゼミβ　第2回

画像処理と移動

今回のゼミ B では DXLIB 内にある新出関数に触れていきます。今回は一部だけ、紹介するのでさらに知りたい方は「http://homepage2.nifty.com/natupaji/DxLib/dxfunc.html」または「DXLIB リファレンス」で検索してみてください。インターネットが使用できない場合は DXLIB_VC 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

- ゲームループに使用する関数

  ProcessMessage 関数

  int ProcessMessage(void) ;

  この関数はウィンドウズが閉じられた時や、エラーが発生した時などの確認をするためにあります。この処理は重要なためゲームループするごとに 1 回の頻度で呼び出されます。

  使用例：while(ProcessMessage() == 0)

  この場合、エラーまたはウィンドウズが閉じられた場合は-1 を返すので while 文を抜けます。

  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。引数には調べるキーが入ります。int 型なので戻り値が 1 なら押されていることを意味し、0 なら押されていないことを意味します。

  使用例：

  while(ProcessMessage() == 0 || CheckHitKey(KEY_INPUT_ESCAPE)==0)

  この使用例では、エラー発生またはウィンドウズが閉じられた場合、または ESCAPE キーが押された場合に while 文から抜けます。

- 画像描画に関する関数

LoadGraph 関数

int LoadGraph(char *FileName) ;

この関数は文字列によ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はこのハンドルを使います。

使用例：int a=LoadGraph(“img/chara.png ”);

この使用例では変数 a に、img ファイルにある chara.png という画像情報が入っているという事を表しています。

DrawGraph 関数

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

この関数では左上の(x,y)を起点に LoadGraph 関数で指定した変数によって画像を描画します。int flag は画像背景の透過を使用するかしないかを判断します

使用例：DrawGraph(10 ,10 ,a ,true);

この使用例では画像の左上の起点を座標(10,10)とし、先ほど取り込んだ変数 a の画像情報を表示しています。true は透過処理をさせることを示しています。処理をしない場合は False になります。

ClearDrawScreen 関数

int ClearDrawScreen (void);

先ほどの描画関数で表示した画像をすべて消去します。ゲーム画面を切り替えるために、新しく画像を描画するときなどに使います。

SetDrawScreen 関数

int SetDrawScreen(int DrawScreen);

この関数は表画面か裏画面のどちらに画像を描画するかを指定する関数です。表に表示した場合、画面がちらついて見えることがあります。これを防ぐために普通は裏画面に表示する SetDrawScreen(DX_SCREEN_BACK);を指定します。

ScreenFlip 関数

int ScreenFlip(void);

先ほど使った SetDrawScreen 関数で指定した裏画面での描画を表画面に反映させる関数です。

- 文字描画関数

GetColor 関数

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

RGB の数を入れることで文字関数で使う色コードを出力してくれます。出したい色が知りたい場合は「RGB　色」で調べてみて下さい。

DrawString 関数

int DrawString(int x ,int y ,char *String , int color);

文字の左上、座標(x,y)を起点にして文字を出力する関数です。int color は先ほどの GetColor 関数を使用します。

使用例：DrawString(10 , 20 , ”HELLO” ,GetColor(255 , 255 , 255));

この例では白い色の文字、HELLO を文字の左上、座標(10,20)を起点にして出力することを表しています。

DrawFormatString 関数

int DrawFormatString(int x , int y , int Color ,char *,・・・);

基本的には DrawString 関数と同じですが printf と同じように%d を使用し、変数を表示することができます。

使用例：DrawFormatString(10 , 10 , GetColor(255,255,255) , %d , x);

この例では x の値を白で文字の左上の座標(10,10)を起点に表示します。

SetFontSize 関数

int SetFontSize(int FontSize);

DrawString などの文字の大きさを指定し変更します。引数は文字の大きさを表しています。

使用例：SetFontSize(10);

```
DrawString( 10 , 10 , ”size of 10” , GetColor( 255,255,255 ));
SetFontSize(100);
DrawString( 100 , 10 , ”size of 100” , GetColor( 255,255,255 ));
```

この例では初めの文字は DrawString で 10 のサイズ、2 つ目の文字は DrawString で 100 のサイズで文字を出力しています。

GetHitKeyStateAll 関数

int GetHitKeyStateAll(char *KeyStateBuf);

この関数では文字が入力されているか、されていないかを判断する関数です。この関数は KeyStateBuf が char 型のポインタを指しており、宣言された 256 個の配列のポインタのうち指定されたキーコードのポインタを受け取ります。またそのために、この関数を使う前に char 型の配列 256 個を宣言します。

使用例：char key[256];

```
if(key[KEY_INPUT_A]){
    DrawString( 10 , 10 , “押されてます”, GetColor( 255,255,255 ));
}else{
   DrawString( 10 , 10 , “押されていません”, GetColor( 255,255,255 ));
}
```

この例では A が押されている瞬間”押されてます”、押されてない場合は”押されていません”と出力します。KEY_INPUT_A は DxLib.h で定義されている定数です。

また、ほかの KEY_INPUT_○○も同じように定義されています。KEY_INPUT_○○とキーボードの対応は DXLIB のリファレンスページの CheckHitKey の説明を参照してください。