DXライブラリ講座

2017/6/19(月)/18:30~

HP : http://dxlib.o.oo7.jp/

DXライブラリとは(Windows版)

- DirectXを使ったWindowsデスクトップアプリの開発に必ず付いて回るDirectXやWindows関連のプログラムを使い易くまとめた形で利用できるようにしたC＋＋言語用のゲームライブラリです。(使用する際はC言語の知識だけで大丈夫)
- これによってプログラマーはゲームの本質的なプログラムに専念することが出来ます。かなり本格的なソフト制作からお遊び程度のミニゲーム制作まで幅広くカバーしています！(HPより転記)

DirectXとは①

- Microsoft DirectX（ダイレクトエックス）は、マイクロソフトが開発したゲーム・マルチメディア処理用のAPIの集合である。

- **アプリケーションプログラミングインタフェース**（API、英: Application Programming Interface）とは、ソフトウェアコンポーネントが互いにやりとりするのに使用するインタフェースの仕様である。

DirectXとは②

- **API**とは、ソフトウェアコンポーネントが互いにやりとりするのに使用するインタフェースの仕様である。

- **ソフトウェアコンポーネント**（Software Componentry）は、ソフトウェアシステムの様々な機能を関心の分離によって分割したものである。
- **関心の分離**（かんしんのぶんり、英語: separation of concerns、SoC）とは、ソフトウェア工学において、プログラムを機能面において可能な限り重複がない、複数の機構に明確に分割することをいう。

DirectXとは③

- **API**とは、ソフトウェアコンポーネントが互いにやりとりするのに使用するインタフェースの仕様である。

- **インタフェース**（英: interface）は、ものごとの境界となる部分と、その境界でのプロトコルを指す。
- **プロトコル**または**プロトコール**（英語: protocol）とは、複数の者が対象となる事項を確実に実行するための手順等について定めたもの。日本語の意訳としては「規定」「議定書」「儀典」などがある。

DirectXとは④

- Microsoft DirectXは、マイクロソフトが開発したゲーム・マルチメディア処理用の「ソフトウェアシステムの機能をそれぞれが重複しない形で分割した物」が互いにやりとりするのに使用する「ための規定（ルール）」の仕様である。

- 特に理解する必要はないです。

DXライブラリとは(再記)

- DirectXを使ったWindowsデスクトップアプリの開発に必ず付いて回るDirectXやWindows関連のプログラムを使い易くまとめた形で利用できるようにしたC＋＋言語用のゲームライブラリです。(使用する際はC言語の知識だけで大丈夫)
- これによってプログラマーはゲームの本質的なプログラムに専念することが出来ます。かなり本格的なソフト制作からお遊び程度のミニゲーム制作まで幅広くカバーしています！(HPより転記)

- 要はDirectXを使いやすくしたよってこと。

（補足）ライブラリとは

- **ライブラリ**（英: Library）とは、汎用性の高い複数のプログラムを再利用可能な形でひとまとまりにしたものである。

- ライブラリと呼ぶ時は、それ単体ではプログラムとして作動させることはできない実行ファイルではない場合がある。
- ライブラリは他のプログラムに何らかの機能を提供するコードの集まりと言うことができる。

コードについて

- ここから、実際に使用するコードの話です。

- 全てここに書いてあります⇒http://dxlib.o.oo7.jp/dxfunc.html

コードについて②

```
1   #include "DxLib.h"
2
3   // プログラムは WinMain から始まります
4   int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
5   {
6       if (DxLib_Init() == -1)      // ＤＸライブラリ初期化処理
7       {
8           return -1;          // エラーが起きたら直ちに終了
9       }
10
11      DrawPixel(320, 240, GetColor(255, 255, 255)); // 点を打つ
12
13      WaitKey();              // キー入力待ち
14
15      DxLib_End();                // ＤＸライブラリ使用の終了処理
16
17      return 0;               // ソフトの終了
18  }
19
```

必ず書くもの①

```
1   #include "DxLib.h"
2
3   // プログラムは WinMain から始まります
4   int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
5   {
6       if (DxLib_Init() == -1)      // ＤＸライブラリ初期化処理
7       {
8           return -1;          // エラーが起きたら直ちに終了
9       }
10
11      DrawPixel(320, 240, GetColor(255, 255, 255)); // 点を打つ
12
13      WaitKey();              // キー入力待ち
14
15      DxLib_End();                // ＤＸライブラリ使用の終了処理
16
17      return 0;               // ソフトの終了
18  }
19
```

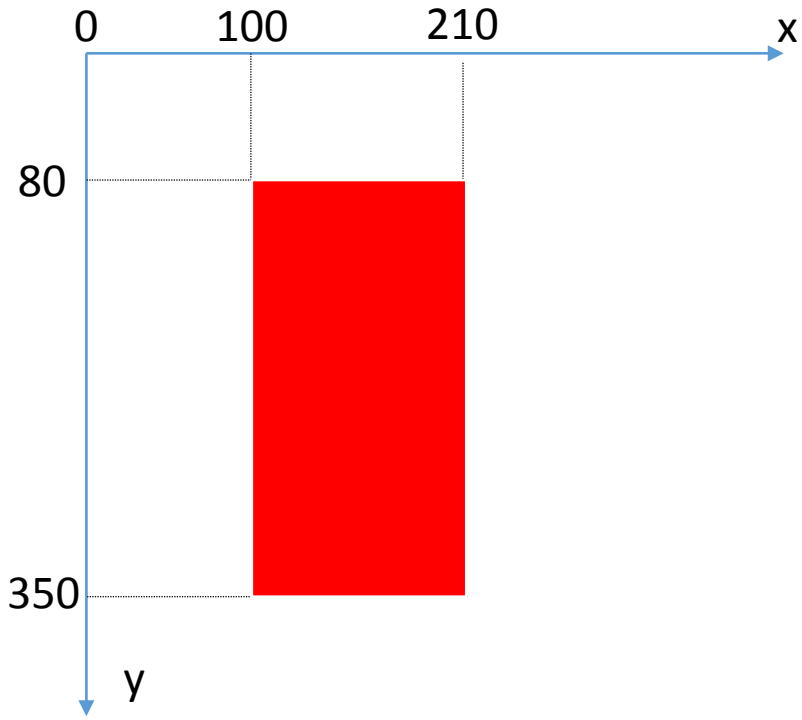
必ず書くもの②

- #include “DxLib.h“
  DXライブラリを使用できるようになる
- int WINAPI WinMain(…) {return 0 ; }
  この中に書いたものが実行される。C言語で言うmain文
- if(DxLib_Init() == -1){ return -1 ; }
  初期処理、最初に必ず行う。
- DxLib_End() ;
  終了処理、最後に必ず行う。

```
1   #include "DxLib.h"
2
3   // プログラムは WinMain から始まります
4   int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
5   {
6       if (DxLib_Init() == -1)     // ＤＸライブラリ初期化処理
7       {
8           return -1;          // エラーが起きたら直ちに終了
9       }
10
11      DrawPixel(320, 240, GetColor(255, 255, 255)); // 点を打つ
12
13      WaitKey();              // キー入力待ち
14
15      DxLib_End();            // ＤＸライブラリ使用の終了処理
16
17      return 0;               // ソフトの終了
18  }
19
```

描画について①

- X軸とY軸の原点に注意。
- 左上が(0,0)、右にx軸が正、下にy軸の正の値を持つ



描画について②(演習)

- 画面に四角形を表示させせよう

  DrawPixel(320 , 240 , GetColor(255,255,255)) ; // 点を打つ

  を

  DrawBox(100 , 80 , 210 , 350 , GetColor(255,0 , 0) , TRUE) ;

  に変更

DrawBox：http://dxlib.o.oo7.jp/function/dxfunc_graph0.html#R2N2

windowモード

- 全画面で起動されて面倒な人向け

- ChangeWindowMode(TRUE);//windowモードに設定
  windowモードに設定する。
- SetGraphMode(x, y, 16);
  windowのサイズを設定する。16はカラービット数。(32も可)

- これらは初期設定に該当するため、DxLib_Init() の前に記述可能

デバッグ用

- int printfDx(char *FormatString , ...) ;

- C言語の標準出力関数である printf 関数と同じような使い方できる。
- 左上に文字の表記が可能。値の確認作業などに使用。

- int clsDx(void) ;
- printfDx()の出力履歴を削除する。上記を使用する場合に併用。

動きの描画について①

- アニメーション(動き)の描画方法

```
#include "DxLib.h" // プログラムはWinMainから始まります

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    SetDrawScreen(DX_SCREEN_BACK);

    if (DxLib_Init() == -1) // ＤＸライブラリ初期化処理
    {
        return -1; // エラーが起きたら直ちに終了
    }

    while (ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0) {
        ClearDrawScreen();// 画面を初期化する

        //ここに描画内容を書く

        ScreenFlip();// 裏画面の内容を表画面に反映させる
    }

    DxLib_End(); // ＤＸライブラリ使用の終了処理
    return 0; // ソフトの終了
}
```

動きの描画について②

- SetDrawScreen(DX_SCREEN_BACK);
  描画は裏画面に行うという設定関数

- ClearDrawScreen();
  表画面をclear(真っ黒)する。
- ScreenFilp();
  裏画面の内容を表に表示。

```
#include "DxLib.h" // プログラムはWinMainから始まります

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    SetDrawScreen(DX_SCREEN_BACK);

    if (DxLib_Init() == -1) // ＤＸライブラリ初期化処理
    {
        return -1; // エラーが起きたら直ちに終了
    }

    while (ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0) {
        ClearDrawScreen();// 画面を初期化する

        //ここに描画内容を書く

        ScreenFlip();// 裏画面の内容を表画面に反映させる
    }

    DxLib_End(); // ＤＸライブラリ使用の終了処理
    return 0; // ソフトの終了
}
```

動きの描画について③

- ProcessMessage()

  定期的に実行する必要あり

  返り値は0(異常無)か-1(異常有)

- CheckHitKey()

  引数のキーが押されているかどうかを返す関数。押されていない=0,押されている=1

  ESCAPEキーを押すことでwhileが終了する仕組み。

```
#include "DxLib.h" // プログラムはWinMainから始まります

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    SetDrawScreen(DX_SCREEN_BACK);

    if (DxLib_Init() == -1) // ＤＸライブラリ初期化処理
    {
        return -1; // エラーが起きたら直ちに終了
    }

    while (ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0) {
        ClearDrawScreen();// 画面を初期化する

        //ここに描画内容を書く

        ScreenFlip();// 裏画面の内容を表画面に反映させる
    }

    DxLib_End(); // ＤＸライブラリ使用の終了処理
    return 0; // ソフトの終了
}
```

動きの描画について④(演習)

- 四角形を動かしてみよう。

1. 描画内容のところにDrawBoxを記入。
2. 表示する座標を変数で指定。
3. while文の中で座標を変更。
   (例 : x = x+1)

```
#include "DxLib.h" // プログラムはWinMainから始まります

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    SetDrawScreen(DX_SCREEN_BACK);

    if (DxLib_Init() == -1) // ＤＸライブラリ初期化処理
    {
        return -1; // エラーが起きたら直ちに終了
    }

    while (ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0) {
        ClearDrawScreen();// 画面を初期化する

        //ここに描画内容を書く

        ScreenFlip();// 裏画面の内容を表画面に反映させる
    }

    DxLib_End(); // ＤＸライブラリ使用の終了処理
    return 0; // ソフトの終了
}
```

キー入力について①

- CheckHitKey()
  引数のキーが押されているかどうかを返す関数。
  押されていない=0,押されている=1

- 例：if(CheckHitKey(KEY_INPUT_RIGHT) == 1){x = x + 1;}
  押している間、常に移動し続ける。

CheckHitKey()：http://dxlib.o.oo7.jp/function/dxfunc_input.html#R5N2

キー入力について②

- 押した時だけ、移動したい場合(1Fだけ処理)

```
int time = 0;
while(…){
	if (CheckHitKey(KEY_INPUT_RIGHT) == 1) {
		time = time + 1;//押された時間
	}
	else if (CheckHitKey(KEY_INPUT_RIGHT) == 0) {//elseでも可
		time = 0;
	}
	if (time == 1) {//押された1Fだけ
		//実行する内容
	}
}
```

キー入力について③(演習)

- 四角形を動かしてみよう。
- 1ページ前のコードを張り付けて、キーを押すと四角形が動くようにする。
- 1Fの実行内容が何かを考える。

```
#include "DxLib.h" // プログラムはWinMainから始まります

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    SetDrawScreen(DX_SCREEN_BACK);

    if (DxLib_Init() == -1) // ＤＸライブラリ初期化処理
    {
        return -1; // エラーが起きたら直ちに終了
    }

    while (ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0) {
        ClearDrawScreen();// 画面を初期化する

        //ここに描画内容を書く

        ScreenFlip();// 裏画面の内容を表画面に反映させる
    }

    DxLib_End(); // ＤＸライブラリ使用の終了処理
    return 0; // ソフトの終了
}
```

画像表示について①

- LoadGraphScreen(int x , int y , char *GraphName , int TransFlag) ;

  GraphNameの画像を、(x,y)の位置で描画する。

  (x,y)の位置が画像の左上の座標になるので注意。

LoadGraphScreen() :
http://dxlib.o.oo7.jp/function/dxfunc_graph1.html#R3N1

```
LoadGraphScreen(100, 200, "C:\\Users\\admin\\Desktop\\img\\photo.jpg" ,true);
```

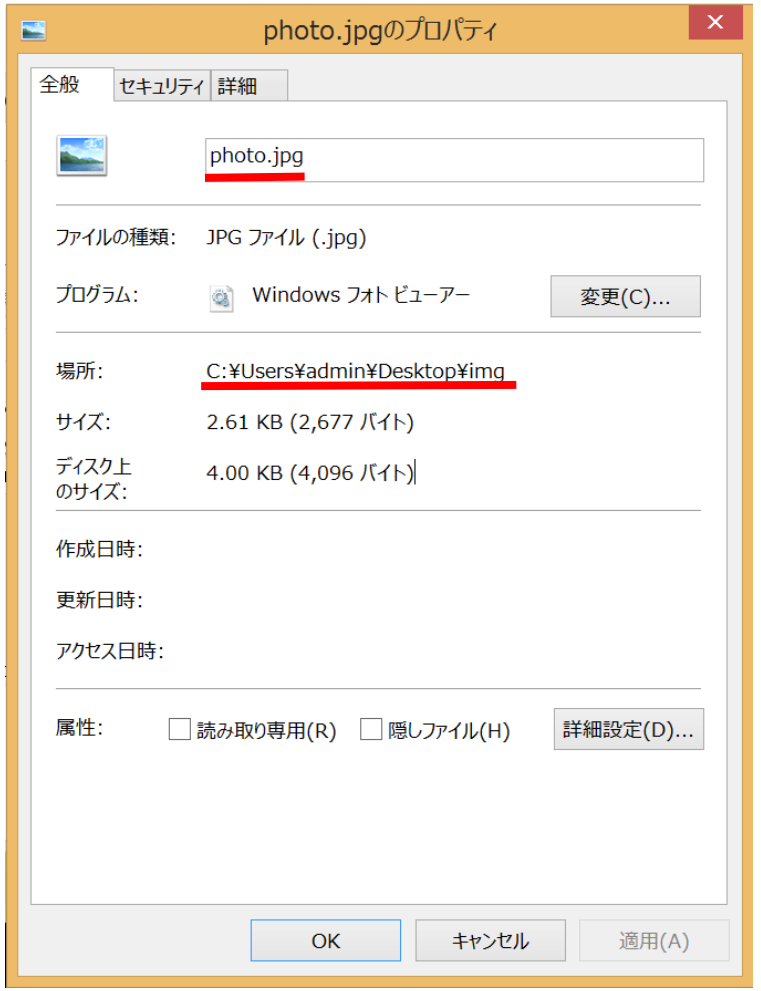
画像表示について②

- char *GraphNameとは
  ファイルのパス(場所)を引数としている。

```
"C:\\Users\\admin\\Desktop\\img\\photo.jpg"
```

  この表記の仕方は絶対パスという。(¥ = ¥¥)

- 相対パスも存在する。



画像表示について③

- 前の画像の表示のさせ方は、処理速度的にはよろしくない。パスの確認と動作確認用。

- int id　= LoadGraph(char *FileName) ;
- int DrawGraph(int x,int y, id, int TransFlag) ;

  変数に読み込んだ画像を数字として保存する。

  画像のパスをメモリに保存することで、処理速度を高速に。

画像表示について④

- 画像サイズについて
- 左上の座標しか指定できないため、きれいに配置しづらい点の解消。

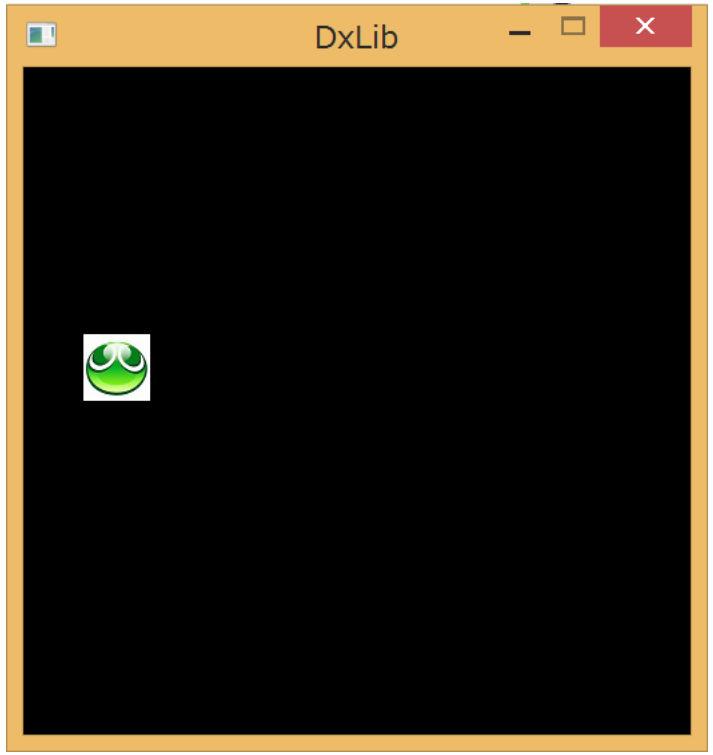
- int GetGraphSize(int GrHandle ,int *SizeXBuf , int *SizeYBuf) ;

```
int id = LoadGraph("C:\\Users\\admin\\Desktop\\img\\photo.jpg");
int size_x,size_y;
GetGraphSize(id, &size_x, &size_y);
DrawGraph(x - size_x / 2, 200 - size_y / 2, id, true);
```

- 変数size_x,size_yにidのサイズがそれぞれ代入される。

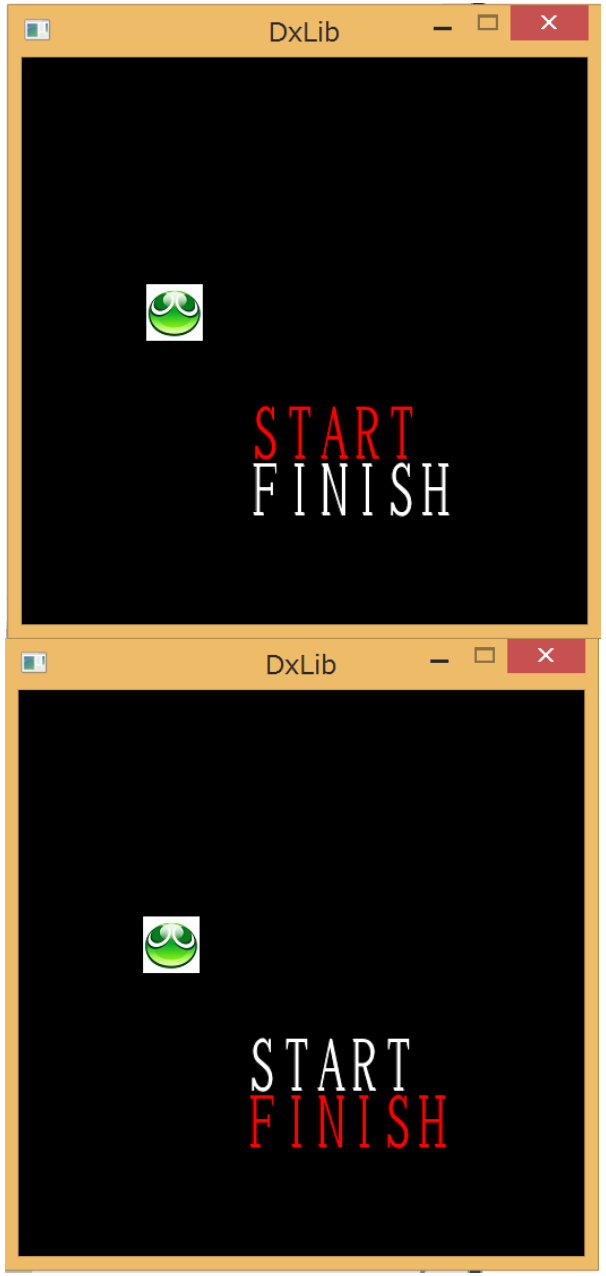
画像表示について⑤(演習)

- 画像を表示させよう。
- 四角形の描画コードを画像に変更(photo.jpg)



画像表示について⑥(演習)

- コマンドを表示させよう。
- 上下キーを押すことで、STARTとFINISHの色が変化するようにする。
- 左右キーで画像が左右に動くよう変更。



本日は以上です。お疲れさまでした。