ソフトゼミ B　第 8 回(アクションゲーム)

DX ライブラリで使える色々な演出

皆様、お疲れ様です。いよいよゼミ B も今回で最後となります。

早速今回の内容についてですが、本日は皆さんにお配りしたアクションゲームに様々な演出を付け加えていただこうと思います。

本題に入る前に先にお伝えさせていただきますが、あくまで今回の内容の大半は応用の範囲ですので、使いこなせないとゲームが作れないというわけではありませんので、知識の一つとして頭に入れておく程度の認識で気楽に取り組んでください(もう既に個人制作に取り掛かってる方はそちらを優先していただいてもかまいません)。

１、BGM の挿入

これから皆さんが個人製作で作るゲームには、必ず BGM を付けていただきます(唐突)。

「以上、ゼミ B 終わり！」と言われても何をどうしたらいいかわからないと思うので、ここで DX ライブラリで音楽を再生する方法を解説します。

DX ライブラリには、指定したファイルの音楽を一行で再生する、PlayMusic(“/*ファイル名*/”,/*再生方法*/)関数と PlaySound(“/*ファイル名*/”,/*再生方法*/)関数というものが存在します。前者は midi 形式の音楽ファイルを再生するときに使用し、後者は wav,ogg,mp3 形式の音楽ファイルを再生するときに使用します(おそらく後者をよく使うことになると思います)。

また、再生方法に関しては、通常再生(DX_PLAYTYPE_NORMAL)・バックグラウンド再生(DX_PLAYTYPE_BACK)・ループ再生(DX_PLAYTYPE_LOOP)の３つのタイプが存在します。用途としては、通常再生やバックグラウンド再生は効果音などの音を一時的に鳴らす場合に使い(通常再生は音楽を流してる間はプログラムの処理が止まるので、基本はプログラムの処理を進めながら音楽を流せるバックグラウンド再生を使った方が事故りにくいです)、ループ再生はステージの BGM などを流すときに使うことになると思います。

音楽を止めたい時には、PlayMusic 関数なら StopMusic()関数、PlaySound 関数なら StopSound()関数を挟むことで音楽を止めることができます。

音楽ファイルを配布するので、実際に自分でプログラムに組み込んでみて動作を確認してみましょう。

※これより下に説明する内容は必ず使う知識ではないので、先に一番後ろの課題１に移っても構いません。

２、画面の明るさとフェードアウト演出

これより説明するのは完全に個人の趣味の範囲なので、力を入れずにそれとなく読んでいただければ十分です。

画面の明るさとありますが、DX ライブラリには **SetDrawBright(/*赤の明るさ*/,/*緑の明るさ*/,/*青の明るさ*/)**関数といったものが存在します。赤・緑・青の明るさには、0～255 の整数値を入れることができ、その数値が小さいほど画面は暗くなります(普段の何もしてない状態の明るさの数値は 255 で、0 になると画面が真っ暗になります)。

具体的な使い方ですが、明るさを調節する関数(例えば int bright＝255 等)を用意し、予めゲームループの中に **SetDrawBright(bright,bright,bright)**関数を置いておき(場所は基本どこでもいいですが、ゲームループの while 文の直下に置くのが望ましいです)、フェードアウトやフェードインを挟みたい場面で bright の数値をいじるのがいいと思います。

ちなみに、ゲームループの while 文は一秒間で約 60 回(個人差があります)回るので、これを参考に明るさの変化する時間の長さを調節してみてください。

また、画面が暗くなってる間にはプレイヤーの操作ができなくなるよう、プレイヤーの移動関数の実行条件に if(bright==255)みたいに付け加えておくと、変なバグが起こりにくくなると思います。

３、キャラクターを歩かせよう

はいそこ、「は？もう動いてんじゃん、ガ〇ジかな？」とか真実を言わない。

確かに、現在のキャラクターの描画方法でもプレイヤーの進行方向に対応して描画がされています。しかし、実際に動いているところを想像してみてください。今のような平行移動だと味気ないですよね？

そこで、一方向に進むキャラクターにも一工夫加えて若干の躍動感()を与えてみましょう。今回お配りした以下の 3 つの画像を使ってもらいます。

player1.png

player2.png

player3.png

心なしか横長に見えるのはきっと気のせいなので、あまり気にしないでください。

さて、皆さんが今まで使ってた画像はplayer2.png(player.png)の一枚だけでした。しかし、今回は三枚の画像を利用してキャラクターを動かしてみようと思います。その一例として、以下のような描画方法があります。

```
int movingright=0;
ーーーーーーーゲームループ開始ーーーーーーーー
if(keyState[KEY_INPUT_RIGHT])
  movingright++;
if(movingright>31)
  movingright=0;
if(movingright<=7||movingright>15&&movingright<=23)
  DrawGraph(player.x,player.y,img.player2,TRUE);
if(movingright>7&&movingright<=15)
  DrawGraph(player.x,player.y,img.player3,TRUE);
if(movingright>23&&movingright<=31)
  DrawGraph(player.x,player.y,img.player1,TRUE);
```

ここでやっていることは、右に進むキーを押している時間をカウントするmovingrightという関数を用意し、そのカウントの値によってplayer2→player3→player2→player1→(最初に戻る)といった具合で画像を変化させています(間隔は8にしてますが、個人のお好みで変えてもらって構いません)。パラパラ漫画を想像してもらうとわかりやすいと思います。

今回の説明は以上となります。習うより慣れた方が使いこなせるようになるのも早いので、バックアップをとったら思う存分にプログラムをいじくりまわしてみてください！

ささやかではありますが、今回教えた内容に関連した課題を以下に書き残しておくので、挑戦してみてください。

課題

1　タイトルからプレイ中の画面に移ったらbgm.mp3を再生し、クリアーかゲームオーバーで再生を中止する

2　ゲームオーバー時にプレイヤーの操作をできなくし、描画を固定したままフェードア

ウト。フェードインでゲームオーバー画面表示

3　左右どちらの移動に対しても、プレイヤーが歩いているかのように描画する

ヒント

1　ゲームループ内で音楽を再生するということを忘れずに！ただ PlaySound 関数を置くだけでは上手く行きません。

2　(フェードアウト中またはフェードイン中)＝(bright!=255)

3　方向を変えたときにカウントはそのままでいいのか・・・？