ソフトゼミＢ　第５回　ファイル分割

今までは一つのファイルにすべてのソースを書いていました。しかし、そうすると内容が多すぎて管理が大変です。なので、ファイルの管理をよりやりやすくするためファイル分割を行いましょう。
（実際の作業としては３ページ目の「今日の作業」と書かれているところからとなります。今回の作業は多少時間がかかるかもしれないので先に作業をやって後で確認することをお勧めします。）

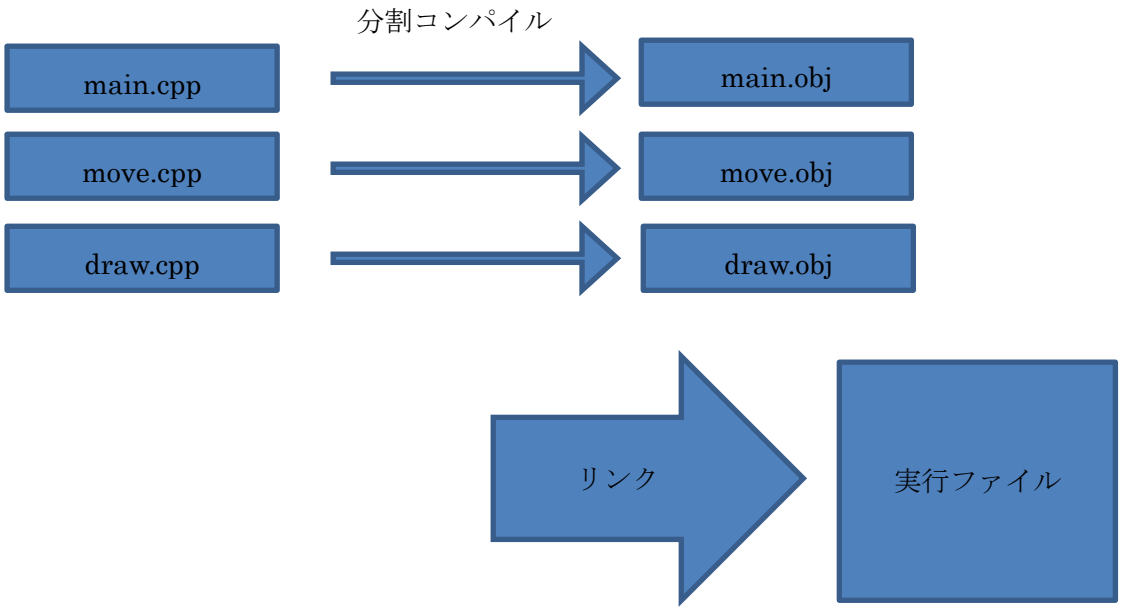
ファイル分割とは？

ファイル分割とは文字通りソースを複数のファイルに分けて書くことです。こうすることのメリットはまずどのプロセスをどこに書いたのかを分かりやすくすることです。たとえば今あるソース内の to_enemy 関数に問題があることがデバッグで分かったとします。これを直そうとしたとき長い文を読んでいくことになり非常に不便です。しかしある程度分類された比較的短い文から探すのは容易です。また、ファイル分割をする目的として複数人で制作するとき役割分担をしやすくなる利点があります。ファイルを分割する方法としては.c または.cpp ファイルを複数用意して分割コンパイルしてリンクする方法とヘッダーファイルを用意する方法があります。

分割コンパイルとリンク

分割コンパイルの過程の説明をします。例として今回作っているゲームのソースを分解する場合を考えます。まず今までは main.cpp のみを使っていましたがそれ以外に自機と敵のの移動をつかさどる関数を格納する move.cpp、画面描画をつかさどる関数を格納する draw.cpp、プログラム内の初期化をつかさどる関数を格納する init.cpp、弾の設置、移動をつかさどる関数を格納する shot.cpp、当たり判定をつかさどる関数を格納する atari.cpp を作って分割コンパイルした場合を考えます。まず、上の六つのファイルを別々にコンパイルします(borland の場合 bcc32 の後に空白を置いて-c を入力してからファイル名入力)。するとコンパイルしたファイル名の拡張子が.obj になったものが出力されます。それらをまとめてコンパイルするとすべてのファイルのソースを網羅した実行ファイルが出力されます。(borland の場合 bcc32 の後に空白を置いて-e を書いた後にそれぞれ空白を置い

て.obj ファイル名（実は.c や.cpp でも可）を書きます。）この作業をリンクといいます。
※Visual C++のプロジェクトでは分割コンパイルとリンクを自動でやってくれます。



ヘッダーファイル

ファイル分割の方法としてヘッダーファイルを使う方法があります。これはあるファイルをコンパイルするときヘッダーファイル内の情報をソースファイルの情報に足し合わせることができます。足し合わせ方としてはソースファイルの先頭に#includeと書いてヘッダー後ろにファイル名を書きます。（ここで気づく人もいると思いますがいつも書いている“stdio.h”、”math.h”などはコンパイラ内で定義されているヘッダーファイルです。またこれらを標準ライブラリといいます。）この時ファイル名を標準ライブラリは<>で囲うのに対して自作ヘッダーは""で囲みます。

extern 宣言

分割コンパイルした場合それぞれのファイルで使用されている変数は共有されてません。たとえば main.cpp 内でグローバル変数で宣言された変数 x はそのままでは move.cpp 内の関数では使えません（move.cpp をコンパイルする時点でエラー）。また、move.cpp 内で普

通にグローバル変数 x を定義しなおしても不具合ができます（リンクをする時点で警告)。理由としてはグローバル変数を定義するときにメモリー領域を割り充てるのですが、先のようにやると x を２回定義したことにより同名の変数が２個できてしまうことにあります。つまり片方のファイルでのみ変数を定義し、それ以外のファイルではそういう名前の変数があることを知らせるだけにしておく必要があります。(これを変数を宣言するといいます。）そこで変数を宣言だけする場合変数の形名の前に extern と書きます。

```
extern 型 変数名;  //宣言のみ
```

今日の作業

Visual C++は分割コンパイルとリンク（２ページ目参照）を自動でやってくれるのですがそれなりの手順を踏まなければなりません。以下に書いていきます。

①左上の新しい項目をクリック

②ソースファイルを作る場合 C++ファイルを、ヘッダーファイルを作る場合ヘッダーファイルをクリックして名前を入力して追加をクリック。作るファイルは以下の通りです。(順番は何でもいいです。）

- ソースファイル
  - init.cpp
  - move.cpp
  - draw.cpp
  - shot.cpp
  - atari.cpp
- ヘッダーファイル
  - define.h
  - struct.h
  - grobal.h

extern.h

function.h

以上です。

③それぞれのファイルに打ち込みます。(コピペしたらいいかもです。) main.cpp も改変します。(内容が長いのでソースファイルは中に入れる関数名を書いておきます。実際に打ち込むときは内容もちゃんと書いてください。)

ヘッダーファイル説明&内容

- define.h
  マクロを入れておくファイルです。

```
#define ENEMY_BURRET 256
#define PLAYER_BURRET 256
#define ENEMY 10
#define HEIGHT  480
#define WIDTH 640
```

- struct.h
  構造体の宣言を入れておくファイルです。

```
#include "define.h"
struct Player
{       double x,y;
        double vx,vy;
         int life;
        double size_x,size_y;
};
struct Enemy
{       double x,y;
        double vx,vy;
        int life;
        int size_x,size_y;
```

```
};
struct Tama
{       double x,y;
        double vx,vy;
        double life;
        double size_x,size_y;
};
struct Item
{       double x,y;
        double vx,vy;
        double life;
        double size_x,size_y;
        int type;

};
struct Img
{
        int player;
        int enemy[10];
        int player_tama;
        int enemy_tama;
        int gameover;
        int title;
        int item;
        int tokuten_tama;
        int haikei;
};
```

- grobal.h
  グローバル変数の定義を入れておくファイルです。

```
#include "struct.h"
struct Player player;
struct Enemy enemy[ENEMY];
struct Tama player_tama[PLAYER_BURRET];
```

```
struct Tama enemy_tama[ENEMY][ENEMY_BURRET];
struct Item item[ENEMY];
struct Item tokuten_tama[ENEMY][ENEMY_BURRET];
struct Img imglist;
 struct Item haikei;
int time;
int tokuten;
char keystate[256];
```

- extern.h
  グローバル変数の宣言を入れておくファイルです。

```
#include "struct.h"
extern struct Player player;
extern struct Enemy enemy[ENEMY];
extern struct Tama player_tama[PLAYER_BURRET];
extern struct Tama enemy_tama[ENEMY][ENEMY_BURRET];
extern struct Item item[ENEMY];
extern struct Item tokuten_tama[ENEMY][ENEMY_BURRET];
extern struct Img imglist;
extern struct Item haikei;
extern int time;
extern int tokuten;
extern char keystate[256];
```

- function.h
  関数のプロトタイプ宣言を入れておくファイルです。

```
int init(void);
void time_progress(void);
void create_enemy(void);
void move_enemy(void);
void enemy_shot(void);
void move_enemy_burret(void);
void move_player(void);
```

```
void player_shot(void);
void move_player_burret(void);
void load(void);
void move_haikei(void);
void to_enemy(void);
void to_player(void);
void player_enemy(void);
void draw(void);
```

ソースファイル説明&内容

- main.cpp
  ゲームの仕組みを書いておくファイルです。

```
#include "Dxlib.h"
#include <math.h>
#include "global.h"
#include "function.h"
int WINAPI WinMain( HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR
lpCmdLine, int nShowCmd ){
ChangeWindowMode( TRUE ) ;
SetGraphMode( WIDTH, HEIGHT, 32 );
init();
load();
if(init()==-1){return-1;}
while(ProcessMessage()==0){
        GetHitKeyStateAll(keystate);
        ClearDrawScreen();
         if(keystate[KEY_INPUT_ESCAPE]){break;}
        DrawGraph(0,0,imglist.title,TRUE);
        if(keystate[KEY_INPUT_RETURN])
        {
                while(ProcessMessage()==0){
                        if(keystate[KEY_INPUT_ESCAPE]){break;}
                        time_progress();
```

```
                        to_enemy();
                        to_player();
                        player_enemy();
                        ClearDrawScreen();
                        draw();
                         if(player.life<=0)
                        {
                                if(keystate[KEY_INPUT_X]){
                                        init();
                                        break;
                                }
                        }

                }
        }
        ScreenFlip();
}
DxLib_End();
return 0;
}
```

- init.cpp
  プログラムの初期化関係の関数を入れておくファイルです。

```
#include "DxLib.h"
#include "extern.h"
#include "function.h"
//関数名だけ書くので定義を書いてください。
int init(void){
…
}
void load(void){
…
}
```

- draw.h
  描画関係の関数を入れておくファイルです。

```
#include "DxLib.h"
#include "extern.h"
#include "function.h"
void draw(){
…
}
```

- shot.h
  弾の発射と移動を制御する関数を入れておくファイルです。

```
#include "DxLib.h"
#include "extern.h"
#include "function.h"
void enemyshot(void){
…
}
void move_enemy_burret(void)
{
…
}
void player_shot(void)
{
…
}
void move_player_burret(void)
{
…
}
```

- atari.cpp
  当たり判定に関係のある関数を入れておくファイルです。

```
#include "DxLib.h"
#include "extern.h"
#include "function.h"
void to_enemy(void)
{
…
}
void to_player(void)
{
…
}
void player_enemy(void)
{
…
}
```

以上です。お疲れ様でした。余談ですがヘッダーファイルを include した場合はその部分をヘッダーファイルの内容に置き換えて考えるとトレースしやすいと思います。