

新規GameObjectの作り方

目次

目次

新規GameObjectを作ろう

まだ作らないでね！！

よし、作っていいぞ！！

メンバ変数

メンバ関数

virtual int GetGameScene(void)override{return GAME_SCENE;}

GameFramework.h

GameFramework.cpp

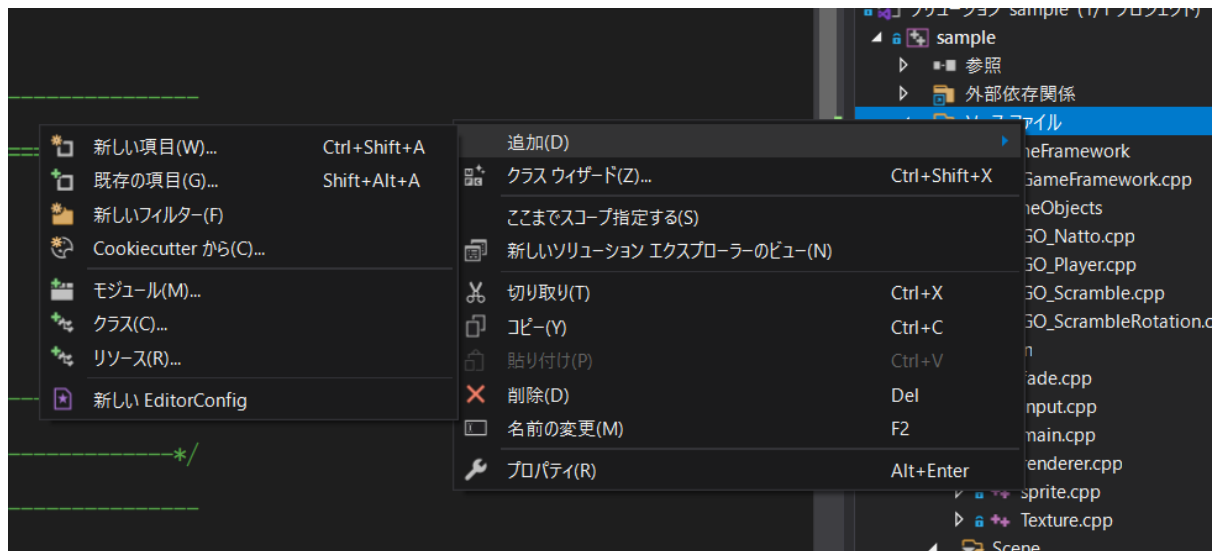
GameSceneの話

GAMESCENE_SCRANBLEの場合

GAMESCENE_BUNGEE_JUMPの場合

ゲームシーンの処理

新規GameObjectを作ろう



↑からクラス (C)...をクリック

クラスの追加

クラス名(L)	.h ファイル(F)		.cpp ファイル(P)	
GO_Throw	GO_Throw.h	...	GO_Throw.cpp	...

クラス名を入力すると.h、.cppに自動で入力されます

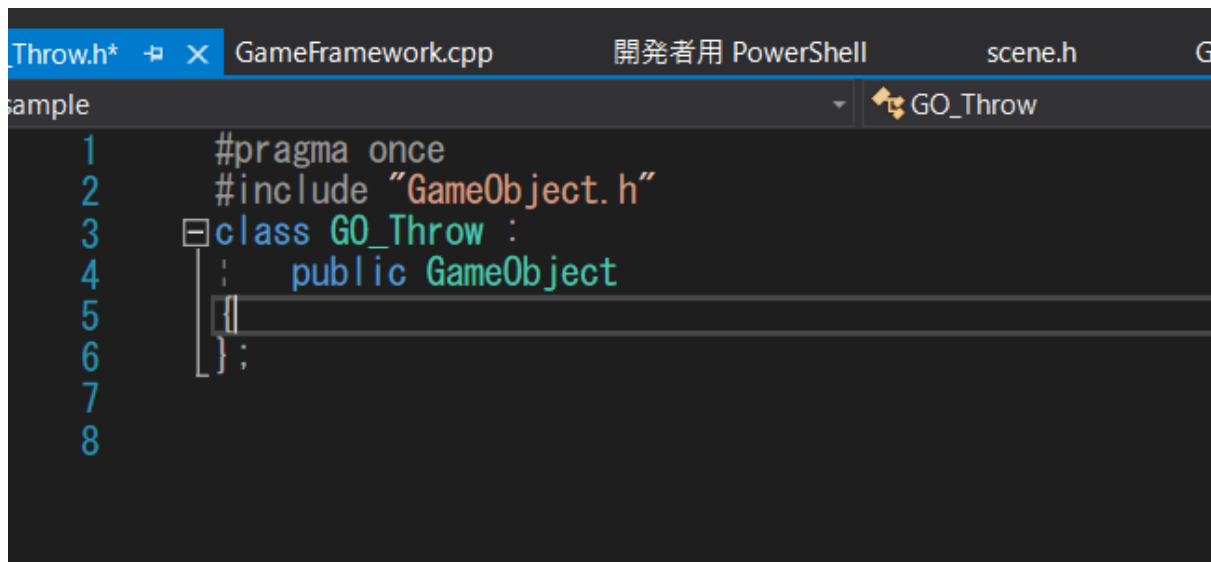
まだ作らないでね！！！！

基底クラス(B)	アクセス(A)
GameObject	public

基底クラスにGameObjectと打ってGameobjectクラスを継承させます

よし、作っていいぞ！！！！

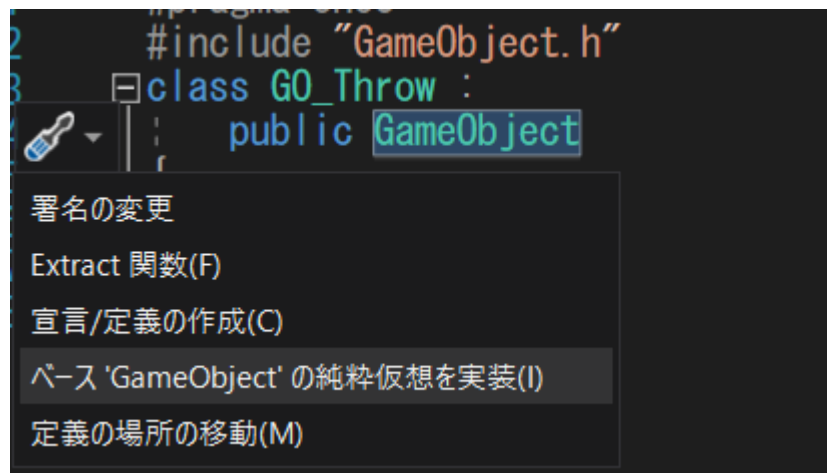
OKボタンを押して作成！



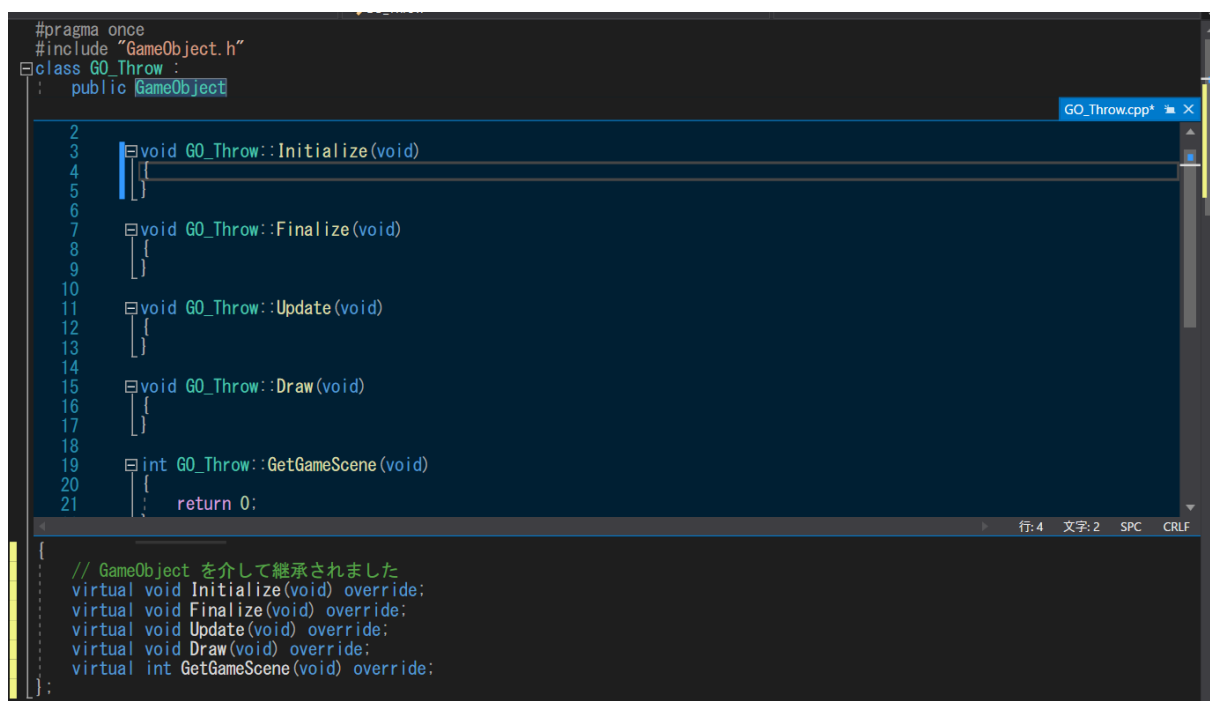
```
1  #pragma once
2  #include "GameObject.h"
3  class GO_Throw :
4  |   public GameObject
5  |   {
6  |   }
7  };
8
```

GameObjectで右クリック！

クイックアクションとリファクタリングを選んで



ベース'GameObject'の純粋仮想を実装をお願いします



こうなったら成功！

メンバ変数

```
private:
    //Texture用のインデックス
    int Throw_Texture;

    //テクスチャの名前
    char TEX_NAME[128] = "data/TEXTURE/";

    //ゲームシーン    -> GameScene.hを参照
    const int GAME_SCENE = GAMESCENE_SCRAMBLE;
};
```

private:

にとりあえず、3つのメンバ変数を作ります。

普段の.cpp内で**グローバル変数**として使っていたものは全部**private**に入れてください。

他で使いたい場合、セッターを作しましょう。

//Texture用のインデックス

int ○○○_Texture;

//テクスチャの名前

char TEX_NAME[128] = "data/TEXTURE/○○○.png";

//ゲームシーン -> GameScene.hを参照

const int GAME_SCENE = GAMESCENE_SCRAMBLE;

メンバ関数

```
virtual int GetGameScene(void)override{return
GAME_SCENE;}
```

```
virtual void Draw(void) override;  
virtual int GetGameScene(void) override { return GAME_SCENE; }
```

Draw関数の下にいるので↑のように内容を変更してね。

.cpp内のやつ消さないとエラー出ます。

これはインラインで書いてください。

```
    {  
    }  
  
    int G0_Throw::GetGameScene(void)  
    {  
        return 0;  
    }
```

＊【余談】 このほうがカッコいいよねってことで変えてます。

- Init() → Initialize()
- Uninit() → Finalize()

一通り終わったら、GameFramework.hに行きます

GameFramework.h

まずは前方宣言

```
//-----前方宣言
class GameObject;
class Enemy;
class GO_Player;
class Collision;
class GO_Scramble;
class GO_ScrambleRotation;
class GO_Throw;
```

次にポインタを作ります

```
private:
    static const int GAME_OBJECT_MAX = 100;

    //GameObjectのポインタ
    GameObject* m_pGameObjects[GAME_OBJECT_MAX];

    GO_Player* mp_player;
    GO_Scramble* mp_vortex;
    GO_ScrambleRotation* mp_VoRot;
    GO_Throw* mp_Throw;

    //GameScene
```

とりあえず、.hはおしまい。.cppに行きます

GameFramework.cpp

```
#include "GameFramework.h"
#include "GameObject.h"
#include "GO_Player.h"
#include "GO_Scramble.h"
#include "GO_ScrambleRotation.h"
#include "GO_Throw.h"
```

新しいGOの.hを作りましょう

▼ .h内にインクルードすりゃええやん

前方宣言をなんのためにやるのかは忘れたけど、.hではあくまでポインタの宣言だけをするために前方宣言してます。

つまり、.h内で処理が行われていないことの証明になります。しらんけど。

Create()を見ていきましょう

```
void GameFramework::Create()
{
    //null
    {
        mp_player = nullptr;
        mp_vortex = nullptr;
        mp_VoRot = nullptr;

        /*ここで新しいゲームオブジェクトのポインタを初期化する*/
    }

    //new
    {
        mp_player = new GO_Player;
        mp_vortex = new GO_Scramble;
        mp_VoRot = new GO_ScrambleRotation;

        /*ゲームオブジェクト動的生成*/
    }

    //Register
    {
        Register(mp_vortex);
        Register(mp_player);
        Register(mp_VoRot);

        /*GameObjectクラスを継承してれば、登録できます。*/
    }
}
```

やるべきことは書いてあるので、同じように追加していきましょう

```

void GameFramework::Create()
{
    //null
    {
        mp_player = nullptr;
        mp_vortex = nullptr;
        mp_VoRot = nullptr;

        /*ここで新しいゲームオブジェクトのポインタを初期化する*/

        mp_Throw = nullptr;
    }

    //new
    {
        mp_player = new GO_Player;
        mp_vortex = new GO_Scramble;
        mp_VoRot = new GO_ScrambleRotation;

        /*ゲームオブジェクト動的生成*/

        mp_Throw = new GO_Throw;
    }

    //Register
    {
        Register (mp_vortex);
        Register (mp_player);
        Register (mp_VoRot);

        /*GameObjectクラスを継承してれば、登録できます。*/

        Register (mp_Throw);
    }
}

```

これで一応動くと思います。

GameSceneの話

ただ、GameFramework.hのm_GameSceneがGAMESCENE_SCRAMBLEで設定されています。


```
GO_THROW* mp_THROW;

//GameScene
int m_GameScene = GAMESCENE_SCRAMBLE;

//nullptr代入→newで動的生成→Register登
```

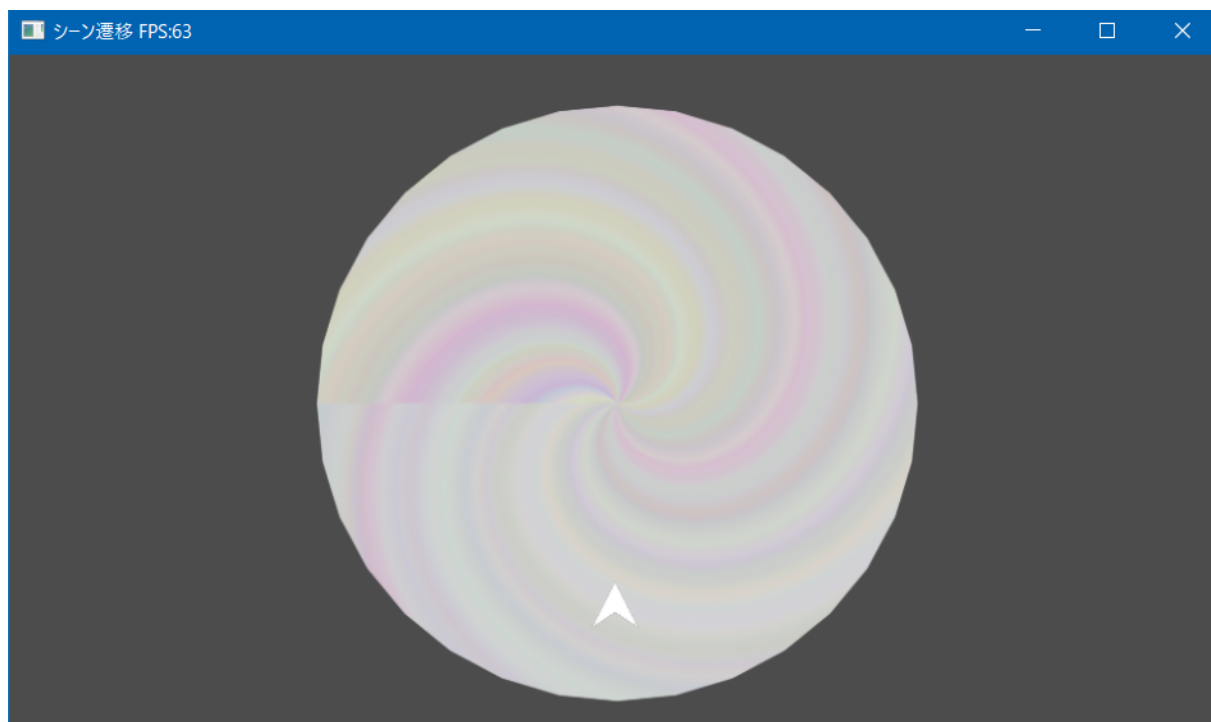
これを変えれば、他のゲームシーンに変えることができます。

GAMESCENE_SCRAMBLEの場合

```
GO_THROW* mp_THROW;

//GameScene
int m_GameScene = GAMESCENE_SCRAMBLE;

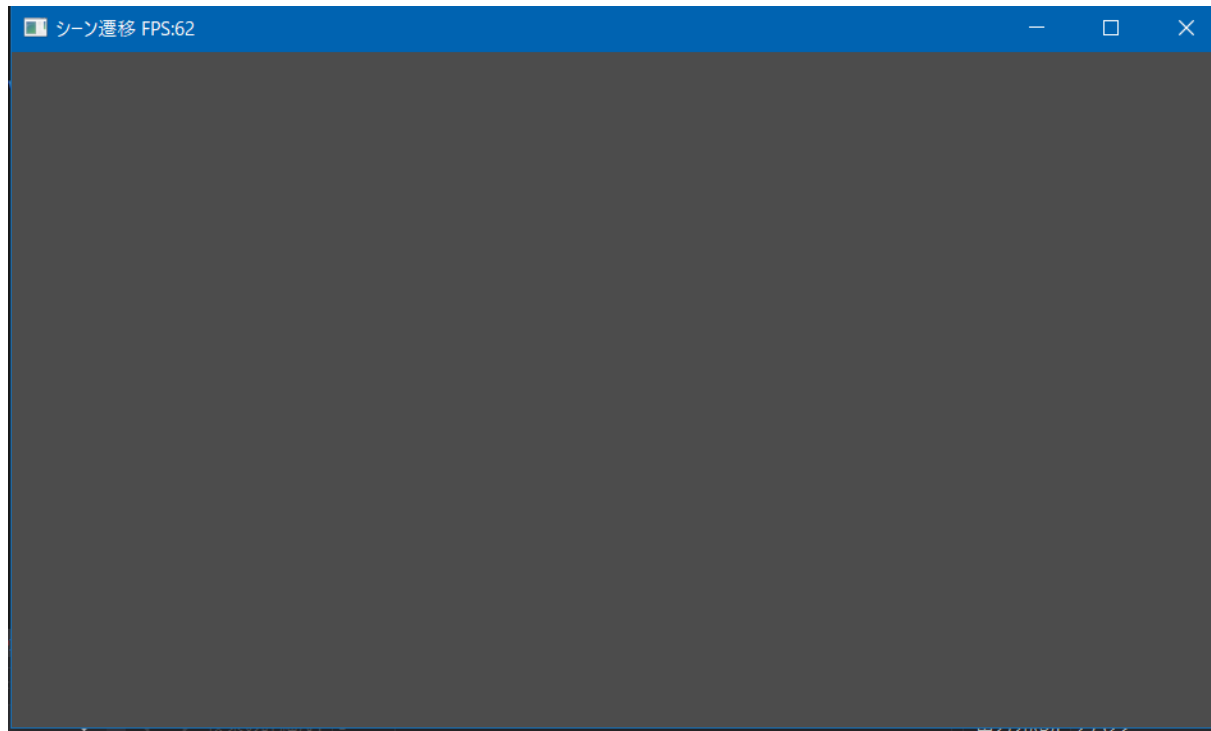
//nullptr代入→newで動的生成→Register登
```



↑既にGO_PlayerやGO_Scrambleが入っている

GAMESCENE_BUNGEE_JUMPの場合

```
//GameScene  
int m_GameScene = GAMESCENE_BUNGEE_JUMP;
```



↑現時点では何も入れてない

ゲームシーンの処理

```

}
void GameFramework::Update(void)
{
    //GameScene.h参照してください
    for (int i = 0; i < GAME_OBJECT_MAX; i++) {
        if (!m_pGameObjects[i]) continue;
        if (m_pGameObjects[i]->GetGameScene() != m_GameScene) continue;

        m_pGameObjects[i]->Update();
    }
}

void GameFramework::Draw(void)
{
    //GameScene.h参照してください
    for (int i = 0; i < GAME_OBJECT_MAX; i++) {
        if (!m_pGameObjects[i]) continue;
        if (m_pGameObjects[i]->GetGameScene() != m_GameScene) continue;
        m_pGameObjects[i]->Draw();
    }
}

```

みて分かる通り、GameFramework内のm_GameSceneとゲームオブジェクトのGAME_SCENEを確認しています。

違ったらその処理は行われません

まだ遷移するための処理を作っていないので、ここで変えてください。