

クラス (class) について ～基本～

C++言語はC言語をオブジェクト指向に拡張したものとよく言われます。オブジェクト指向とは、一つ一つの部品 (Builder では Label、Shape などのコンポーネント) を「物」と考えてプログラムを組み立てることです。

class とはそのオブジェクト指向プログラミングにおいて、データとその操作手順であるメソッドをまとめたオブジェクトの雛型を定義したものです。Builder の Shape などのコンポーネントもあらかじめ定義されている class です。

どういうことかよくわからなかったら、最初は「class とは構造体の拡張版だ」と思うと良いかも知れません。実際に使ってみると、class は構造体とよく似ているところが多々あります。

実際に class 「物」を定義してみましょう。ここでは例として「アクションゲームのキャラクター」(以降キャラクター) という class をつくります。

キャラクターには、「HP」「位置座標」という要素を持ち、「歩く」「ジャンプする」などの動作ができます。プログラミングでは前者をメンバ、後者をメソッドと呼びます。

これを書くときは、Builder の「ファイル→新規作成→ヘッダーファイル」を選択し、新たなファイルに書きましょう。これと同時に「ファイル→新規作成→cpp ファイル」を作っておくと便利です。

二つとも同じファイル名で Project1 と同じところに保存しましょう。

できたら新規のヘッダーにクラスを宣言します。

```
D:\user_Lilyy-B\編江春\class実験\class.h
Unit1.cpp | Unit1.h | class.cpp | class.h
#include <vcl.h> //Stringを使うために必要
class TActChara //アクションゲームのキャラクター
{
public:
String name; //キャラクターの名前
int HP; //キャラクターのHP
int Left,Top,Width,Height; //キャラクターの位置座標
void Appear(); //キャラクターの登場
void Walk(); //歩くという動作
};
10: 49 変更あり 挿入
```

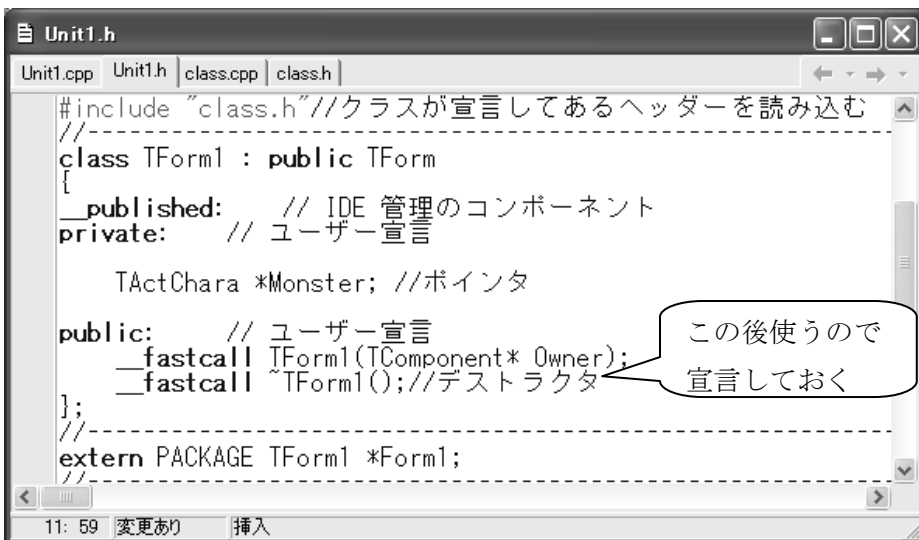
これでクラスの宣言はできました。やっていることは構造体とほぼ一緒です。

次にメソッドの内容を書いていきます。ヘッダーファイルに直接書くのも良いですが、先ほど作ったcppファイルに書くと見やすくなります。

```
class.cpp
Unit1.cpp | Unit1.h | class.cpp | class.h
#include "class.h" //クラスが宣言してあるヘッダーを読み込む
//-----
void TActChara::Appear()
{
//キャラクターの登場
/*
処理
*/
}
//-----
void TActChara::Walk()
{
//キャラクターが歩く動作
/*
処理
*/
}
//-----
18: 78 変更あり 挿入
```

これでクラスの準備は完了です。次は実際に宣言してみましょう。

Unit1.hにクラスが宣言してあるファイルを読み込み、クラスのポインタを宣言します。



```
#include "class.h"//クラスが宣言してあるヘッダーを読み込む
//-----
class TForm1 : public TForm
{
__published: // IDE 管理のコンポーネント
private: // ユーザー宣言

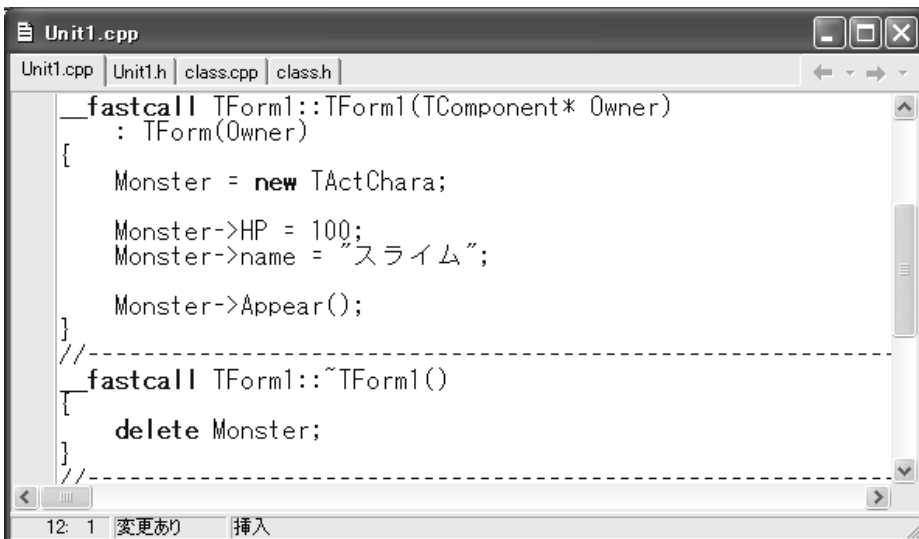
    TActChara *Monster; //ポインタ

public: // ユーザー宣言
    __fastcall TForm1(TComponent* Owner);
    __fastcall ~TForm1(); //デストラクタ
};
//-----
extern PACKAGE TForm1 *Form1;
//-----
```

この後使うので
宣言しておく

11: 59 変更あり 挿入

これで、TActChara のオブジェクト（インスタンスとも呼ぶ）Monsterのポインタができました。これが使えるように Unit1.cpp で new しましょう。この時デストラクタで delete するのを忘れないでください。(new、delete についてはこの冊子の new、delete を参照してください。)



```
__fastcall TForm1::TForm1(TComponent* Owner)
: TForm(Owner)
{
    Monster = new TActChara;

    Monster->HP = 100;
    Monster->name = "スライム";

    Monster->Appear();
}
//-----
__fastcall TForm1::~~TForm1()
{
    delete Monster;
}
//-----
```

12: 1 変更あり 挿入

あとは Label や Shape と同じように、クラスのメンバに変数を代入するときは

```
Monster->Left = 150;
```

のように、Shape などと同じようにアロー演算子「->」で指定すればいいのです。

ここまでは、クラスの宣言やそのオブジェクトの使用を説明してきました。ここまでだと構造体でもできるのですが、これだけでもプログラムの効率がよくなります。

また、自分で作ったオブジェクトがいままで使ってきたコンポーネントと同じ用につかえるので、よりプログラムの組み立てがわかりやすかとおもいます。

他にクラスができることの中に、継承といわれるものがあります。これはクラスの機能拡張を簡単にできるものです。使いこなせれば大変便利なものになります。

これについては「クラス (class) について ~継承~」をご覧ください。

