オブジェクト指向と ゲームプログラミング

▶ コンポーネント編 ― 第４回　ビットマップクラスの作成 ◀

Strategyパターン

Strategyパターンは、デザインパターンの１つで、各アルゴリズムをカプセル化して、それらを交換可能にするというものです。アルゴリズムだけ交換したいときは、純粋仮想関数を持つ基底クラスを作成し、それをさまざまなアルゴリズムで実装した派生クラスを用意すれば、アップキャストにより実行時に切り替えることができます。これがStrategyパターンです。

Strategyパターンのメリットは、アルゴリズムと、その利用側を独立させることで、アルゴリズムを自由に変更することができるという点と、どのアルゴリズムを使用するのかをプログラム実行時に選択することができるという点が挙げられます。

課　題

CDDBクラスとCDIBクラスは、ビットマップオブジェクトを管理するという共通の目的があります。また、これらのクラスは、インタフェース(メンバ関数の戻り値や型)もビットマップ生成関数以外は、ほとんど同じです。そこで、共通点をTemplate Methodパターンでまとめて基底クラスCBitmapを作成します。DDBとDIBで異なる部分は、Strategyパターン(とStateパターン)で実装します。

こうすると、DDBとDIBが交換可能になり、さらに２つの違いをほとんど意識することなく、扱うことができます。

(１)ビットマップオブジェクトを管理するクラスCBitmapを作成します。CBitmapクラスのヘッダファイル(Bitmap.hpp)を以下のように作成しましょう。

―Bitmap.hpp―

```
/*
================================================================================================
                                オブジェクト指向ゲームプログラミング
     Programmed by Hibikino software. Copyright (c) 2004 Hibikino software. All rights reserved.
================================================================================================

  【対象ＯＳ】
     Microsoft Windows98/2000以降

  【コンパイラ】
     Microsoft VisualC++ 6.0J ServicePack6

  【プログラム】
     Bitmap.hpp
                ビットマップクラスヘッダ

  【履歴】
     * Version     1.00       2004/04/dd  hh:mm:ss

================================================================================================
*/

#pragma once

/*********************************************************************************************/
/*                                        インクルードファイル                                 */
/*********************************************************************************************/
#include <windows.h>

/*********************************************************************************************/
/*                                      ビットマップクラス定義                                 */
/*********************************************************************************************/
class CBitmap {
public:
    CBitmap();
    virtual ~CBitmap() { Release(); }
```

```
    virtual bool CreateFromBitmap(LPCTSTR inFileName) = 0;

    void Release();
    bool LoadFromFile(LPCTSTR inFileName);
    UINT SetPalette(RGBQUAD inColors[], const UINT inEntries);
    UINT GetPalette(RGBQUAD outColors[]);

    // アクセス関数
    HDC    GetDC()     const { ※ここは各自考えましょう }
    BITMAP GetBitmap() const { ※ここは各自考えましょう }
    LONG   GetWidth()  const { ※ここは各自考えましょう }
    LONG   GetHeight() const { ※ここは各自考えましょう }
    LONG   GetPitch()  const { ※ここは各自考えましょう }
    WORD   GetBPP()    const { ※ここは各自考えましょう }
    LPVOID GetImage()  const { ※ここは各自考えましょう }

protected:
    HDC       m_hDC;
    HBITMAP   m_hBitmap;
    HBITMAP   m_hDefBitmap;
    BITMAP    m_Bitmap;

private:
    CBitmap(const CBitmap&);
    CBitmap& operator=(const CBitmap&);
};
```

(２)CBitmapクラスのソースファイル(Bitmap.cpp)を以下のように作成しましょう。

－Bitmap.cpp－

```
/*
=============================================================================================
                              オブジェクト指向ゲームプログラミング
     Programmed by Hibikino software. Copyright (c) 2004 Hibikino software. All rights reserved.
=============================================================================================

  【対象ＯＳ】
     Microsoft Windows98/2000以降

  【コンパイラ】
     Microsoft VisualC++ 6.0J ServicePack6

  【プログラム】
     Bitmap.cpp
                ビットマップクラス

  【履歴】
     * Version    1.00       2004/04/dd  hh:mm:ss

=============================================================================================
*/

#pragma once

/*********************************************************************************************/
/*                                      インクルードファイル                                 */
/*********************************************************************************************/
#include ※ここは各自考えましょう
#include <olectl.h>

/*********************************************************************************************/
/*                                    デフォルトコンストラクタ                               */
/*********************************************************************************************/
CBitmap::CBitmap() : ※ここは各自考えましょう
{
    ※ここは各自考えましょう;
}

/*********************************************************************************************/
/*                                          解　放                                           */
/*********************************************************************************************/
void CBitmap::Release()
{
```

```
    ※この関数の内容は各自考えましょう
}

/*****************************************************************************************************/
/*                                          イメージファイル読み込み                                 */
/*****************************************************************************************************/
bool CBitmap::LoadFromFile(LPCTSTR inFileName)
{
    ※この関数の内容は各自考えましょう
}

/*****************************************************************************************************/
/*                                          パレット設定                                             */
/*****************************************************************************************************/
UINT CBitmap::SetPalette(RGBQUAD inColors[], const UINT inEntries)
{
    ※この関数の内容は各自考えましょう
}

/*****************************************************************************************************/
/*                                          パレット取得                                             */
/*****************************************************************************************************/
UINT CBitmap::GetPalette(RGBQUAD outColors[])
{
    ※この関数の内容は各自考えましょう
}
```

(３)ビットマップオブジェクトをDDBで管理するクラスCDDBを作成します。基本的な部分はCBitmapクラスで実装されているので、差分のみ実装します。また、純粋仮想関数CreateFromBitmap関数のオーバーライドが必要になります。
　以上をふまえ、以下のCDDBクラスの定義をBitmap.hppに追加しましょう。

```
/*****************************************************************************************************/
/*                                          DDBクラス定義                                            */
/*****************************************************************************************************/
class CDDB : ※ここは各自考えましょう {
public:
    CDDB() {}
    CDDB(const HDC hDC, const int inWidth, const int inHieght) { Create(hDC, inWidth, inHieght); }
    CDDB(const HDC hDC, LPCTSTR inBMPFileName) { CreateFromBitmap(hDC, inBMPFileName); }
    CDDB(LPCTSTR inBMPFileName) { CreateFromBitmap(NULL, inBMPFileName); }
    virtual ~CDDB() {}

    bool Create(const HDC hDC, const int inWidth, const int inHeight);
    bool CreateFromBitmap(const HDC hDC, LPCTSTR inBMPFileName);

    // 純粋仮想関数の実装
    virtual bool CreateFromBitmap(LPCTSTR inBMPFileName) { return ※ここは各自考えましょう; }

private:
    CDDB(const CDDB&);
    CDDB& operator=(const CDDB&);
};
```

(４)CDDBクラスのソースファイル(DDB.cpp)を以下のように作成しましょう。

－DDB.cpp－

```
/*
=================================================================================================
                                オブジェクト指向ゲームプログラミング
    Programmed by Hibikino software. Copyright (c) 2004 Hibikino software. All rights reserved.
=================================================================================================

  【対象ＯＳ】
    Microsoft Windows98/2000以降

  【コンパイラ】
    Microsoft VisualC++ 6.0J ServicePack6

  【プログラム】
    DDB.cpp
                DDB(デバイス依存ビットマップ)クラス
```

```
  【履歴】
    * Version    1.00       2004/04/dd  hh:mm:ss

==========================================================================================
*/

/*****************************************************************************************/
/*                                    インクルードファイル                                 */
/*****************************************************************************************/
#include ※ここは各自考えましょう

/*****************************************************************************************/
/*                                      DDB生成                                          */
/*****************************************************************************************/
bool CDDB::Create(const HDC hDC, const int inWidth, const int inHeight)
{
    ※この関数の内容は各自考えましょう
}

/*****************************************************************************************/
/*                                      DDB生成                                          */
/*****************************************************************************************/
bool CDDB::CreateFromBitmap(const HDC hDC, LPCTSTR inBMPFileName)
{
    ※この関数の内容は各自考えましょう
}
```

(５)ビットマップオブジェクトをDIBで管理するクラスCDIBを作成します。基本的な部分はCBitmapクラスで実装されているので、差分のみ実装します。また、純粋仮想関数CreateFromBitmap関数のオーバーライドが必要になります。
以上をふまえ、以下のCDIBクラスの定義をBitmap.hppに追加しましょう。

```
/*****************************************************************************************/
/*                                      DIBクラス定義                                    */
/*****************************************************************************************/
class CDIB : ※ここは各自考えましょう {
public:
    CDIB() {}
    CDIB(const LONG inWidth, const LONG inHieght, const WORD inBPP) { Create(inWidth, inHieght, inBPP); }
    CDIB(LPCTSTR inBMPFileName) { CreateFromBitmap(inBMPFileName); }
    virtual ~CDIB() {}

    bool Create(const LONG inWidth, const LONG inHieght, const WORD inBPP);

    // 純粋仮想関数の実装
    virtual bool CreateFromBitmap(LPCTSTR inBMPFileName);

private:
    CDIB(const CDIB&);
    CDIB& operator=(const CDIB&);
};
```

(６)CDIBクラスのソースファイル(DIB.cpp)を以下のように作成しましょう。

―DIB.cpp―

```
/*
==========================================================================================
                        オブジェクト指向ゲームプログラミング
    Programmed by Hibikino software. Copyright (c) 2004 Hibikino software. All rights reserved.
==========================================================================================

  【対象ＯＳ】
    Microsoft Windows98/2000以降

  【コンパイラ】
    Microsoft VisualC++ 6.0J ServicePack6

  【プログラム】
    DIB.cpp
                DIB(デバイス独立ビットマップ)クラス
```

```
【履歴】
    * Version    1.00       2004/04/dd  hh:mm:ss

================================================================================================
*/

/*****************************************************************************************/
/*                                    インクルードファイル                                   */
/*****************************************************************************************/
#include ※ここは各自考えましょう

/*****************************************************************************************/
/*                                      DIB生成                                          */
/*****************************************************************************************/
bool CDIB::Create(const LONG inWidth, const LONG inHeight, const WORD inBPP)
{
    ※この関数の内容は各自考えましょう
}

/*****************************************************************************************/
/*                                      DIB生成                                          */
/*****************************************************************************************/
bool CDIB::CreateFromBitmap(LPCTSTR inBMPFileName)
{
    ※この関数の内容は各自考えましょう
}
```

(７)CBitmapクラスが正しく動作するか確認します。CTestSceneクラスを以下のように変更しましょう。

・CTestSceneクラスのメンバに追加

```
CBitmap*   m_pBitmap;
```

・CTestScene::CTestScene関数を以下のように変更

```
CTestScene::CTestScene()
{
    // ここに、機能テストの初期化処理を記述します
    m_pBitmap = new CDDB("BMP\\Hanabi.bmp");

    FPSTimer().Reset();         // タイマリセット
}
```

・CTestScene::~CTestScene関数を以下のように変更

```
CTestScene::~CTestScene()
{
    // ここに、機能テストの解放処理を記述します
    delete m_pBitmap;
}
```

・CTestScene::ActivateProc関数を以下のように変更

```
int CTestScene::ActiveProc()
{
    // ここに、機能テストのメイン処理を記述します
    if(FPSTimer().IsSkip() == false) {
        HDC   hDestDC = ::GetDC(GameApp().GetHWnd());
        ::BitBlt(hDestDC, 0, 0, 640, 480, m_pBitmap->GetDC(), 0, 0, SRCCOPY);
        FPSTimer().DrawFPS(hDestDC);
        ::ReleaseDC(GameApp().GetHWnd(), hDestDC);
    }

    return 0;
}
```

(８)クラス相関図は、次のようになります。