オブジェクト指向と ゲームプログラミング

▶ DirectX Graphics編 － 第３回　デバイスオブジェクト ◀

Direct3DDevice9オブジェクト

Direct3D9オブジェクトが生成できたら、Direct3DDevice9オブジェクトを生成します。このオブジェクトは、グラフィックデバイスを制御したり、Direct3Dのほとんどのオブジェクトを生成するための、いわばDirectX Graphicsの要となるオブジェクトです。

Direct3DDevice9オブジェクトは、プリミティブ、頂点バッファ、シェーダ、サーフェス、ガンマコントロールなどグラフィックデバイスを制御するものから描画まで、かなり多くの機能を提供しています。また、このオブジェクトからほかのオブジェクトを生成または取得したり、Direct3Dエクステンション(D3DXライブラリ)が提供するほとんどの関数で必要になるなど、DirectX Graphicsを使った多く処理で使われます。

Direct3DDevice9オブジェクトの生成は、IDirect3D9::CreateDeviceメソッドで行います。

IDirect3D9::CreateDeviceメソッド

－説明－

CreateDeviceメソッドは、ディスプレイアダプタ(グラフィックデバイス)を管理するDirect3DDevice9オブジェクトを生成し、そのインタフェースを返します。

－書式－

```
HRESULT CreateDevice(UINT Adapter, D3DDEVTYPE DeviceType, HWND hFocusWindow,
                     DWORD BehaviorFlags, D3DPRESENT_PARAMETERS* pPresentationParameters,
                     IDirect3DDevice9** ppReturnedDeviceInterface);
```

－パラメータ－

１つ目の引数(Adapter)は、使用するディスプレイアダプタ(グラフィックデバイス)の番号です。D3DADAPTER_DEFAULTを指定すると、現在使用されているアダプタになります。

２つ目の引数(DeviceType)は、デバイスの種類です。以下のいずれかを指定します。

D3DDEVTYPE_HAL	HALデバイス。ハードウェアによるラスタライズを行います。シェーディングはハードウェアかソフトウェアで行い、座標変換と照明演算はハードウェアかソフトウェアまたはその両方で行います。
D3DDEVTYPE_REF	すべての機能がソフトウェアで実行されるリファレンスラスタライザ。すべての機能が正確に実装されていますが、速度は大幅に低下します。 このデバイスを使用するには、開発者ランタイムをインストールし、リファレンスラスタライザの使用を有効にする必要があります。
D3DDEVTYPE_SW	プラグ可能なソフトウェアデバイス。このソフトウェアはプログラマが用意する必要があります。

３つ目の引数(hFocusWindow)は、フォーカスを設定するウィンドウのハンドルです。メインウィンドウのハンドルを指定します。

４つ目の引数(BehaviorFlags)は、デバイスの動作を示すフラグです。通常は以下のいずれかを指定します。

D3DCREATE_HARDWARE_VERTEXPROCESSING	ハードウェアで頂点処理を行います。
D3DCREATE_SOFTWARE_VERTEXPROCESSING	ソフトウェアで頂点処理を行います。ライト数に制限がなく、バーテックスシェーダがソフトウェアで実行されます。
D3DCREATE_MIXED_VERTEXPROCESSING	ハードウェアとソフトウェアで頂点処理を行います。

５つ目の引数(pPresentationParameters)は、デバイスのパラメータを設定したD3DPRESENT_PARAMETERS構造体変数のアドレスです。このメソッドを呼び出すと、以下のメンバの値が初期化されたデバイスのものに書き換えられます。

ウィンドウモード	BackBufferCount, BackBufferWidth, BackBufferHeight BackBufferFormat
フルスクリーンモード	BackBufferCount, BackBufferWidth, BackBufferHeight

６つ目の引数(ppReturnedDeviceInterface)は、生成されるオブジェクトのインタフェースを格納するIDirect3DDevice9*型またはLPDIRECT3DDEVICE9型の変数のアドレスを指定します。

－戻り値－
成功した場合はD3D_OK、それ以外の場合はエラーコードを返します。

５つ目の引数で必要となるD3DPRESENT_PARAMETERS構造体は、以下のメンバで構成されています。ここで指定した情報に沿ってデバイスが初期化されます。ほとんどのメンバは省略しても(０でも)構いませんが、一部のデバイスでは不具合が報告されているので、すべてのメンバに適切な値を設定しておいた方が安全です。

メンバ	データ型	機能
BackBufferWidth	UINT	バックバッファの幅
BackBufferHeight	UINT	バックバッファの高さ
BackBufferFormat	D3DFORMAT	バックバッファのフォーマット
BackBufferCount	UINT	バックバッファの枚数
MultiSampleType	D3DMULTISAMPLE_TYPE	マルチサンプリングの種類
MultiSampleQuality	DWORD	マルチサンプリングの品質
SwapEffect	D3DSWAPEFFECT	スワップエフェクトの種類
hDeviceWindow	HWND	デバイスと関連づけるウィンドウのハンドル
Windowed	BOOL	ウィンドウモードであるかの真偽値
EnableAutoDepthStencil	BOOL	自動深度ステンシルバッファの真偽値
AutoDepthStencilFormat	D3DFORMAT	自動深度ステンシルバッファのフォーマット
Flags	DWORD	デバイスのフラグ
FullScreen_RefreshRateInHz	UINT	リフレッシュレート
PresentationIntervals	UINT	スワップを実行するタイミング

BackBufferWidth, BackBufferHeight
バックバッファの幅と高さのピクセル数を指定します。ウィンドウモードの場合、０を指定するとhDeviceWindowメンバまたはフォーカスウィンドウ(CreateDeviceメソッドの３つ目の引数)のクライアント領域のサイズになります。フルスクリーンモードでは、この値が画面のサイズになります。

BackBufferFormat
バックバッファのフォーマットを指定します。以下のフォーマットが指定できます。

・ウィンドウモード
D3DFMT_A8R8G8B8　D3DFMT_X8R8G8B8　D3DFMT_A1R5G5B5　D3DFMT_X1R5G5B5　D3DFMT_R5G6B5
・フルスクリーンモード
D3DFMT_X8R8G8B8　D3DFMT_X1R5G5B5　D3DFMT_R5G6B5　D3DFMT_A2R10G10B10

ウィンドウモードでは、D3DFMT_UNKNOWNを指定すると現在使用されているディスプレイモードと同じフォーマットを使用することができます。また、ディスプレイモードと一致しないフォーマットを使用することもできます。この場合、ハードウェアで色変換が行われ、プライマリに転送されます。
フルスクリーンモードでは、この値がディスプレイモードのフォーマットになります。

BackBufferCount
バックバッファの枚数を指定します。０，１，２，３のいずれかを指定します。０は１として扱われます。通常は、０または１を指定します。

SwapEffect
スワップエフェクトの指定です。スワップとは、プライマリサーフェスとバックバッファを交換することです。以下のいずれかを指定します。

D3DSWAPEFFECT_DISCARD	最も効率の良いスワップの方法をディスプレイドライバが選択します。ただし、スワップ後のバックバッファの内容は保証されません。
D3DSWAPEFFECT_FLIP	プライマリサーフェスとバックバッファを常に交換します。
D3DSWAPEFFECT_COPY	バックバッファをプライマリサーフェスにコピーします。スワップ後も、バックバッファの内容が破壊されることはありません。

hDeviceWindow
バックバッファと関連づけるウィンドウのハンドルを指定します。NULLを指定するとフォーカスウィンドウが使用されます。

Windowed
DirectX Graphicsがウィンドウモードで動作するかどうかを指定します。TRUE(非０)を指定するとウィンドウモード、FALSEを指定するとフルスクリーンモードになります。

EnableAutoDepthStencil
DirectX Graphicsが深度バッファ(ピクセルの前後を判断するバッファ)を管理するかどうかを指定します。TRUEを指定するとDirectX Graphicsが深度バッファを管理します。FALSEを指定すると自動管理は行われません。

AutoDepthStencilFormat
DirectX Graphicsが管理する深度バッファのフォーマットを指定します。

```
// プレゼンテーションパラメータ設定(hWndはメインウィンドウのハンドル)
D3DPRESENT_PARAMETERS   d3dpp;
ZeroMemory(&d3dpp, sizeof(d3dpp));
d3dpp.BackBufferCount             = 1;                          // バックバッファ数
d3dpp.MultiSampleType             = D3DMULTISAMPLE_NONE;        // マルチサンプルタイプ
d3dpp.MultiSampleQuality          = 0;                          // マルチサンプル品質
d3dpp.SwapEffect                  = D3DSWAPEFFECT_DISCARD;      // スワップエフェクト
d3dpp.hDeviceWindow               = hWnd;                       // ターゲットウィンドウ
d3dpp.Windowed                    = TRUE;                       // 動作モード
d3dpp.EnableAutoDepthStencil      = TRUE;                       // 深度バッファ
d3dpp.AutoDepthStencilFormat      = D3DFMT_D16;                 // 深度バッファフォーマット
d3dpp.Flags                       = 0;                          // フラグ
d3dpp.FullScreen_RefreshRateInHz = 0;                           // リフレッシュレート

if(d3dpp.Windowed != 0) {
    // ウィンドウモード
    d3dpp.BackBufferWidth      = 0;                             // バックバッファ幅
    d3dpp.BackBufferHeight     = 0;                             // バックバッファ高さ
    d3dpp.BackBufferFormat     = D3DFMT_UNKNOWN;                // バックバッファフォーマット
    d3dpp.PresentationInterval = D3DPRESENT_INTERVAL_IMMEDIATE; // スワップインターバル
} else {
    // フルスクリーンモード
    d3dpp.BackBufferWidth      = 640;                           // バックバッファ幅
    d3dpp.BackBufferHeight     = 480;                           // バックバッファ高さ
    d3dpp.BackBufferFormat     = D3DFMT_X8R8G8B8;               // バックバッファフォーマット
    d3dpp.PresentationInterval = D3DPRESENT_INTERVAL_DEFAULT;   // スワップインターバル
}

// Direct3DDevice9オブジェクト生成(pD3Dは初期化済みのDirect3D9オブジェクト)
IDirect3DDevice9*   pD3DDevice = NULL;
pD3D->CreateDevice(D3DADAPTER_DEFAULT, D3DDEVTYPE_HAL, hWnd,        // hWndはウィンドウのハンドル
                   D3DCREATE_MIXED_VERTEXPROCESSING, &d3dpp, &pD3DDevice);
```

Direct3DDevice9オブジェクトの解放

Direct3DDevice9オブジェクトの解放は、ほかのオブジェクトと同じようにReleaseメソッドで行います。

```
// デバイスオブジェクト解放
pD3Device->Release();
```

課　題

CDXGraphics9クラスに、Direct3DDeviceオブジェクトを生成および解放する処理を追加しましょう。

(1)Initialize関数の引数を以下のように変更しましょう(プロトタイプと関数本体の２カ所変更します)。

```
bool Initialize(const HWND hWnd, const bool inWindowed);
```

Direct3DDevice9オブジェクトの生成は、Initialize関数で行います。このオブジェクトを生成するときに、メインウィンドウのハンドルが必要になります。これは、Initialize関数を呼び出すときに引数で渡してもらいます。また、ウィンドウモードかどうかを表すフラグ(inWindowed)も引数として追加しています。trueならウィンドウモード、falseならフルスクリーンモードで初期化を行います。これらのほかにも初期化時に必要な情報で、かつ外部のみでしか取得できない情報がある場合は、この例を参考に引数に追加してください。

(2)プレゼンテーションパラメータを管理するメンバを追加します。以下のプログラムを適切な場所に追加しましょう。

```
D3DPRESENT_PARAMETERS   m_PresentParams;
```

(３)プレゼンテーションパラメータを初期化する処理を追加します。以下のプログラムをコンストラクタとRelease関数の適切な場所に追加しましょう。

```
::ZeroMemory(&m_PresentParams, sizeof(m_PresentParams));
```

(４)プレゼンテーションパラメータのうち、ウィンドウモードとフルスクリーンで共通の値となる部分(一度設定したら書き換える必要のないもの)を設定します。以下のプログラムをInitialize関数の適切な場所に追加しましょう。

```
// プレゼンテーションパラメータ ― 共通部設定
m_PresentParams.BackBufferCount           = 1;                                  // バックバッファ数
m_PresentParams.MultiSampleType           = D3DMULTISAMPLE_NONE;                // マルチサンプルタイプ
m_PresentParams.MultiSampleQuality        = 0;                                  // マルチサンプル品質
m_PresentParams.SwapEffect                = D3DSWAPEFFECT_DISCARD;              // スワップエフェクト
m_PresentParams.hDeviceWindow             = hWnd;                               // ターゲットウィンドウ
m_PresentParams.EnableAutoDepthStencil    = TRUE;                               // 深度バッファ
m_PresentParams.AutoDepthStencilFormat    = D3DFMT_D16;                         // 深度バッファフォーマット
m_PresentParams.Flags                     = D3DPRESENTFLAG_LOCKABLE_BACKBUFFER; // フラグ
m_PresentParams.FullScreen_RefreshRateInHz = 0;                                 // リフレッシュレート
```

(５)プレゼンテーションパラメータの残りの部分(ウィンドウモードとフルスクリーンモードで異なる部分)を設定する処理を追加します。この処理は、ウィンドウモードとフルスクリーンモードを切り替えるときにも必要となるので、関数にしておきます。
以下のプロトタイプをprivateに追加しましょう。

```
void SetPresentParams(const BOOL inWindowed);
```

この関数は、CDXGraphics9クラスの内部でしか使用しません。外部から自由に使用できてしまうと、DirectX Graphicsの実際の状態とプレゼンテーションパラメータの値が異なるといった矛盾が生じてしまう可能性があります。このように、外部から使用すると不具合が起こるような関数は、privateにしておき隠蔽しておきます。

(６)SetPrasentParams関数を実装します。以下のプログラムをソースファイルに追加しましょう。

```
/*****************************************************************************************************/
/*                                プレゼンテーションパラメータ設定                                   */
/*****************************************************************************************************/
void CDXGraphics9::SetPresentParams(const BOOL inWindowed)
{
    if(inWindowed != FALSE) {
        // ウィンドウモード
        m_PresentParams.Windowed             = TRUE;                           // 動作モード
        m_PresentParams.BackBufferWidth      = 0;                              // バックバッファ幅
        m_PresentParams.BackBufferHeight     = 0;                              // バックバッファ高さ
        m_PresentParams.BackBufferFormat     = D3DFMT_UNKNOWN;                 // バックバッファフォーマット
        m_PresentParams.PresentationInterval = D3DPRESENT_INTERVAL_IMMEDIATE;  // スワップインターバル
    } else {
        // フルスクリーンモード
        m_PresentParams.Windowed             = FALSE;                          // 動作モード
        m_PresentParams.BackBufferWidth      = 640;                            // バックバッファ幅
        m_PresentParams.BackBufferHeight     = 480;                            // バックバッファ高さ
        m_PresentParams.BackBufferFormat     = D3DFMT_X8R8G8B8;                // バックバッファフォーマット
        m_PresentParams.PresentationInterval = D3DPRESENT_INTERVAL_DEFAULT;    // スワップインターバル
    }
}
```

(７)以下の処理を(４)の直後に追加しましょう。

```
// プレゼンテーションパラメータ ― 非共通部設定
SetPresentParams(inWindowed);
```

これで、プレゼンテーションパラメータのすべてのメンバが設定されたので、Direct3DDevice9オブジェクトを生成することができます。

(８)CDXGraphics9クラスに、Direct3DDevice9オブジェクトを管理するメンバ変数を追加します。以下のプログラムを適切な場所に追加しましょう。

```
IDirect3DDevice9*   m_pD3DDevice;
```

(９)CDXGraphics9クラスのインスタンス生成時に、m_pD3DDeviceメンバに初期値が設定されるようにします。コンストラクタに以下の初期化子を追加しましょう。

```
CDXGraphics9::CDXGraphics9() : m_pD3D(NULL), m_pD3DDevice(NULL)
{
}
```

(10)Direct3DDevice9オブジェクトの生成を行うプログラムを追加します。Initialize関数の適切な場所に、以下のプログラムを完成させて追加しましょう。

```
// Direct3DDevice9オブジェクト生成
const DWORD   Behavior[3] = {D3DCREATE_MIXED_VERTEXPROCESSING,
                             D3DCREATE_HARDWARE_VERTEXPROCESSING,
                             D3DCREATE_SOFTWARE_VERTEXPROCESSING};
HRESULT       hr;
for(int i = 0; i < 3; i++) {
    hr = ※ここは各自考えましょう;
    if(hr == D3D_OK)
        break;
}
if(hr != D3D_OK) {
    ::OutputDebugString("*** Error - Direct3DDevice9オブジェクト生成失敗(CDXGraphics9_Initialize)\n");
    Release();
    return false;
}
```

(11)Direct3DDevice9オブジェクトの解放処理を、Direct3D9オブジェクトの解放を参考に作成し、Release関数の適切な場所に追加しましょう。

(12)CGameApp::Initialize関数の「DirectX Graphicsの初期化」を以下のように変更しましょう。

```
// DirectX Graphics初期化
if(DXGraphics().Initialize(GetHWnd(), inWindowed) == false)
    return false;
```