ゲームプログラミング

▶　DirectX Audio－第２回　DirectX Audioのオブジェクト　◀

DirectX Audioの基本的なオブジェクトの説明と、それらの生成と解放について解説します。

DirectX Audioの基本的なオブジェクト

DirectX Audioにはさまざまなオブジェクトがあります。オーディオファイルを読み込んで再生するだけでも、以下のオブジェクトを使用します。

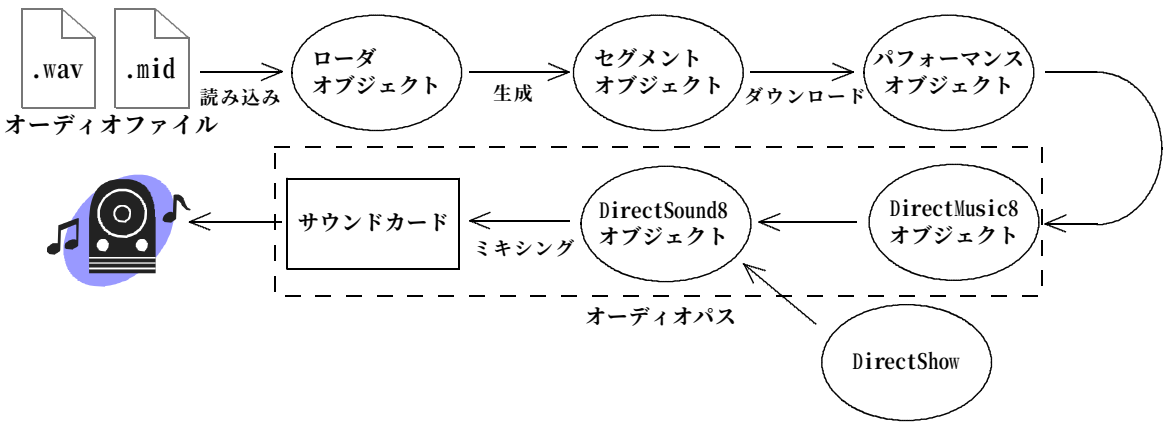
DirectMusic8オブジェクト
DirectMusicのもっとも基本的なオブジェクトです。ほとんどの場合、内部的に作成され、暗黙的に使用されるので、このオブジェクトを生成する必要はありません。

DirectSound8オブジェクト
DirectSoundのもっとも基本的なオブジェクトです。DirectX Audioでは、オーディオデータの管理はDirectMusicだけで十分に行えますが、DirectShowとミキシングしたり、サウンドデータが格納されているメモリ領域(バッファ)にアクセスするような高度な操作を行う場合は、このオブジェクトが必要になります。このオブジェクトも内部的に生成され、暗黙的に使用されます。

DirectMusicLoader8オブジェクト
ローダオブジェクトは、オーディオファイルの検索や読み込みに使用します。

DirectMusicPerformance8オブジェクト
パフォーマンスオブジェクトオブジェクトは、演奏全体に関わる情報を管理します。オーディオデータの経路(オーディオパス)の設定、セグメントオブジェクトの再生状態などを管理します。

DirectMusicSegment8オブジェクト
セグメントオブジェクトは、オーディオファイルを管理します。オーディオファイルは、ローダオブジェクトによってセグメントオブジェクトに読み込まれます。セグメントオブジェクトをパフォーマンスオブジェクトにダウンロードすることにより、再生できるようになります。



CoCreateInstance関数

DirectX Audioの一部のオブジェクトは、CoCreateInstance関数で生成します。

CoCreateInstance関数
－説明－
CoCreateInstance関数は、指定されたクラス参照識別子に関連づけられたオブジェクトを生成し、そのインタフェースを取得します。

－パラメータ－
１つ目の引数は、生成されるオブジェクトに関連づけられたクラスの参照識別子(クラスID)です。

２つ目の引数は、通常は使用しないのでNULLにします。

３つ目の引数は、生成されるオブジェクトの種類(コンテキスト)です。DirectX Audioのオブジェクトを生成する場合は、"CLSCTX_INPROC"を指定します。

４つ目の引数は、生成されるオブジェクトのインタフェース参照識別子(インタフェースID)です。

５つ目の引数は、生成されるオブジェクトのインタフェースを格納する変数のアドレスです。

－戻り値－
関数が成功するとS_OK、それ以外はエラーの原因をエラーコードで返します。

COMライブラリの初期化と解放

CoCreateInstance関数のように、先頭に"Co"の文字がつくものはCOMライブラリの関数です。このような関数は、COMライブラリを初期化してから呼び出さないと必ず失敗します。COMライブラリの初期化は、CoInitialize関数を呼び出すことで行います。

```
// COMライブラリ初期化
CoInitialize(NULL);
```

COMライブラリの関数を使用する必要がなくなった場合は、必ずCoUninitialize関数を呼び出し、COMライブラリが使用した資源を解放します。

```
// COMライブラリ解放
CoUninitialize();
```

SUCCEEDEDマクロとFAILEDマクロ

DirectXの関数やメソッドでは、戻り値として「成功を示す値」と「失敗を示す値」のほかに、「成功ではないが成功と見なしてもかまわない値」を返す場合があります。
メソッドの呼び出しが成功すると、D3D_OK, DI_OK, S_OKといった成功を示す値が返されます。メソッドの中には、これら以外の値でも成功とみなしてもかまわない値を返す場合があります。このような値を含めた成否を判定するために、SUCCEEDEDマクロとFAILEDマクロが提供されています。どちらのマクロも引数に関数やメソッドの戻り値を与えます。
SUCCEEDEDマクロは、関数やメソッドの戻り値が成功または成功とみなしてもかまわない値なら真を返します。FAILEDマクロはその逆で、メソッドの戻り値が成功を示す値でも成功とみなしてもかまわない値でもないとき(すなわち、完全に失敗のとき)に真を返します。

```
if(S_OK == CoCreateInstance(省略)) {
    // S_OKのみ成功
}

if(S_OK != CoCreateInstance(省略)) {
    // S_OK以外は失敗(成功と見なしてもかまわない値も失敗と見なす)
}

if(SUCCEEDED(CoCreateInstance(省略))) {
    // 関数呼び出しが成功
}

if(FAILED(CoCreateInstance(省略))) {
    // 関数呼び出しが失敗
}
```

DirectXでは、成功とみなしてもかまわない値を返すメソッドが多くあるので、すべてのメソッド呼び出しでSUCCEEDEDマクロかFAILEDマクロで成否を判定した方がよいでしょう。

ローダオブジェクトの生成

ローダオブジェクトは、CoCreateInstance関数で生成します。１つ目の引数はローダオブジェクトのクラスID"CLSID_DirectMusicLoader"、２つ目の引数はNULL、３つ目の引数は"CLSCTX_INPROC"、４つ目の引数はローダオブジェクトのインタフェースID"IID_IDirectMusicLoader8"、５つ目の引数は、生成されるローダオブジェクトのインターフェイスを格納する変数(IDirectMusicLoader8*型)のアドレスをLPVOID*型にキャストして指定します。

```
IDirectMusicLoader8*   pIDMLoader8 = NULL;   // ローダオブジェクト
CoCreateInstance(CLSID_DirectMusicLoader, NULL, CLSCTX_INPROC,
                 IID_IDirectMusicLoader8, (LPVOID*)&pIDMLoader8);
```

オブジェクトの操作

DirectXのオブジェクトは、すべてC++のオブジェクトと同じように、メソッドを呼び出す形式で行います。従来のAPIはC言語の関数と同じ形をしています。たとえば、仮にDirectMusicLoaderを解放するDirectMusicLoaderRelease関数があるとすれば、

```
DirectMusicLoaderRelease(pIDMLoader8);  // pIDMLoader8は初期化済みのローダオブジェクト
```

のように、操作対象を引数で渡す形式になっています。DirectXは、C++のクラスオブジェクトのような形式になっているため、上記の関数呼び出しは、すべてメソッド呼び出しになります。

```
pIDMLoader8->Release();                 // pIDMLoader8は初期化済みのローダオブジェクト
```

操作対象であるオブジェクトから"->"演算子でメソッドを呼び出します。C++はこのように、オブジェクトに備えられたメソッドを呼び出して操作します。本来、オブジェクトのメソッドを呼び出す場合、構造体と同じように"."(ドット)演算子を使うので、

```
pIDMLoader8.Release();                  // pIDMLoader8は初期化済みのローダオブジェクト
```

とするべきですが、DirectXのオブジェクトへの操作はすべて、インタフェースのアドレス(すなわちポインタ)を通して間接的に行うので、"->"演算子でメソッドを呼び出します。なお、メソッドとはオブジェクトに備えられた関数で、メンバ関数と呼ばれることもあります。

ローダオブジェクトの解放

生成または取得したDirectXオブジェクトは、プログラム終了までに必ず解放しなければなりません。すべてのオブジェクトには解放するためのReleaseメソッドがあり、これを呼び出すと解放されます。解放されたオブジェクトは再び初期化するまで使用できないので、オブジェクトを格納していた変数には無効を表すNULLを代入し、無効なオブジェクトであることを明示するようにします。このようしておくと、解放したオブジェクトを初期化なしに再使用してしまい、不安定な動作になってしまうというバグを防ぐことができます。

```
// ローダオブジェクト解放
if(NULL != pIDMLoader8) {       // ポインタチェック。NULLの場合は解放しない(する必要がない)
    pIDMLoader8->Release();     // ローダオブジェクト解放
    pIDMLoader8 = NULL;         // NULLを代入し、無効なオブジェクトであることを示す
}
```

パフォーマンスオブジェクトの生成

パフォーマンスオブジェクトは、CoCreateInstance関数で生成します。１つ目の引数はパフォーマンスオブジェクトのクラスID"CLSID_DirectMusicPerformance"、２つ目の引数はNULL、３つ目の引数は"CLSCTX_INPROC"、４つ目の引数はパフォーマンスオブジェクトのインタフェースID"IID_IDirectMusicPerformance8"、５つ目の引数は、生成されるパフォーマンスオブジェクトのインターフェイスを格納する変数(IDirectMusicPerformance8*型)のアドレスをLPVOID*型にキャストして指定します。

```
IDirectMusicPerformance8*   pIDMPerformance8 = NULL; // パフォーマンスオブジェクト
CoCreateInstance(CLSID_DirectMusicPerformance, NULL, CLSCTX_INPROC,
                 IID_IDirectMusicPerformance8, (LPVOID*)&pIDMPerformance8);
```

オーディオパスの初期化

オーディオパスとは、オーディオデータがどのような経路をとおってスピーカーに到達するかを定義するものです。たとえば、MIDIデータは外部に接続されているMIDI音源を使うのか、それともソフトウェアシンセサイザを使うのかを定義する必要があります。Waveデータはどのサウンドカードを使って再生するのかを決める必要があります。

オーディオパスはかなり詳細に定義できますが、通常はデフォルトのオーディオパスで十分です。デフォルトオーディオパスでは、MIDIデータは、ソフトウェアシンセサイザ「Microsoft GS Wavetable Synth」でWaveデータに変換されます。Waveデータは、DirectSoundによりほかのデータとミキシングされ、デフォルトのサウンドカードによってスピーカーに出力されます。

デフォルトオーディオパスを初期化するのがパフォーマンスオブジェクトのInitAudioメソッドです。

InitAudioメソッド
－説明－
InitAudioメソッドは、パフォーマンスを初期化し、デフォルトオーディオパスを使用可能にします。

－パラメータ－
１つ目の引数は、DirectMusicオブジェクトのインタフェースを指定または取得する場合に使用します。NULLを指定すると暗黙的に使用されます。

２つ目の引数は、DirectSoundオブジェクトのインタフェースを指定または取得する場合に使用します。NULLを指定すると暗黙的に使用されます。

３つ目の引数は、DirectSoundオブジェクトの初期化で使用するメインウィンドウのハンドルです。NULLを指定すると、現在前面にあるウィンドウが使用されます。

４つ目の引数は、オーディオパスのタイプを指定します。おもに以下のものを使用します。

DMUS_APATH_DYNAMIC_MONO	モノラル出力
DMUS_APATH_DYNAMIC_STEREO	ステレオ出力
DMUS_APATH_SHARED_STEREOPLUSREVERB	ステレオ出力とリバーブ効果
DMUS_APATH_DYNAMIC_3D	３Ｄサウンド

５つ目の引数は、パスに割り当てるパフォーマンスのチャンネル数(同時可能演奏数)です。

６つ目の引数は、要求する機能を指定します。通常は"DMUS_AUDIOF_ALL"を指定し、すべての機能が使用できるようにします。

７つ目の引数は、シンセサイザ(データを演奏する機能)のパラメータを指定します。通常は使用しないのでNULLにします。

－戻り値－
成功した場合はS_OK、それ以外はエラーの原因をエラーコードで返します。

```
IDirectMusicPerformance8*   pIDMPerformance8;          // 初期化済みのパフォーマンスオブジェクト
pIDMPerformance8->InitAudio(NULL, NULL, hWnd, DMUS_APATH_DYNAMIC_STEREO,
                            64, DMUS_AUDIOF_ALL, NULL); // hWndはウィンドウのハンドル
```

パフォーマンスオブジェクトの解放

パフォーマンスオブジェクトの解放は、CloseDownメソッドでオブジェクトを閉じたあと、Releaseメソッドで解放します。

```
IDirectMusicPerformance8*   pIDMPerformance8;     // パフォーマンスオブジェクト(生成済みとする)
pIDMPerformance8->CloseDown();
pIDMPerformance8->Release();
```

○ 課題 ○

ローダオブジェクトとパフォーマンスオブジェクトの生成と解放処理を追加しましょう。

(１)COMライブラリの初期化処理を適切な場所に追加しましょう。COMライブラリはDirectShowなど、DirectX Audio以外でも必要になるので、DXAInit関数以外の適切な場所で初期化します。

－追加１－
```
// COMライブラリ初期化
CoInitialize(NULL);
```

(２)COMライブラリの解放処理を適切な場所に追加しましょう。COMライブラリはDirectShowなど、DirectX Audio以外でも使用しているので、DXARelease関数以外の適切な場所で解放します。
－追加２－
```
// COMライブラリ解放
CoUninitialize();
```

(３)以下のプログラムを適切な場所に追加しましょう。

－追加３－
```
static IDirectMusicLoader8*       g_pIDMLoader8      = NULL;
static IDirectMusicPerformance8*  g_pIDMPerformance8 = NULL;
```

(４)ローダオブジェクトの生成、パフォーマンスオブジェクトの生成、デフォルトオーディオパスの初期化を行う以下のプログラムを完成させ、適切な場所に追加しましょう。

－追加４－
```
    // ローダオブジェクト生成
    if(FAILED(CoCreateInstance(??????????????????????, NULL, CLSCTX_INPROC,
                               ??????????????????????, (LPVOID*)&g_pIDMLoader8))) {
        OutputDebugString("*** Error - ローダオブジェクト生成失敗(DXAInit)\n");
        return false;
    }

    // パフォーマンスオブジェクト生成
    if(FAILED(※ここは各自考えましょう)) {
        OutputDebugString("*** Error - パフォーマンスオブジェクト生成失敗(DXAInit)\n");
        DXARelease();
        return false;
    }

    // デフォルトオーディオパス初期化
    if(S_OK != ※ここは各自考えましょう(※ステレオ出力かステレオ出力＋リバーブで初期化すること)) {
        OutputDebugString("*** Error - デフォルトオーディオパス初期化失敗(DXAInit)\n");
        DXARelease();
        return false;
    }
```

(５)ローダオブジェクトとパフォーマンスオブジェクトの解放を行う以下のプログラムを完成させ、適切な場所に追加しましょう。

－追加５－
```
    // パフォーマンス解放
    if(NULL != g_pIDMPerformance8) {
        g_pIDMPerformance8->??????????();    // パフォーマンスオブジェクトを閉じる
        g_pIDMPerformance8->???????();       // パフォーマンスオブジェクトを解放
        g_pIDMPerformance8 = NULL;
    }

    // ローダ解放
    if(NULL != g_pIDMLoader8) {
        g_pIDMLoader8->????????();
        g_pIDMLoader8 = NULL;
    }
```