▶ DirectX Audio－第１２回　DirectSound3Dと３Ｄバッファの生成 ◀

DirectSound3Dでは、サウンドに位置と速度を持たせることができます。ステレオでは表現できない立体的なサウンドや、3Dグラフィックと組み合わせた臨場感のある仮想空間を表現することができます。

DirectSound3D

DirectSound3Dでは、音源(3Dサウンドバッファ)は、3D空間の座標を持ちます。また、音を聞くオブジェクトのを再現したリスナーという概念も存在します。
左右のスピーカから出力される音量は、DirectSound3Dが音源とリスナーの座標から計算します。そのため、ステレオデータは意味を持ちません。
音源、リスナーともに位置と速度を持ちます。リスナーが音源に近づくほど音量が大きくなります。速度が設定されている場合、ドップラー効果が自動的に計算されます。ただし、この速度は計算のための値であり、音源やリスナーが自動で動くというわけではありません。位置はプログラムで適切な値を設定します。
また、DirectSound3Dでは、D3DVALUEやD3DVECTORといったDirect3Dでよく使われる型が使われるほか、座標系も同じ左手系であるため、Direct3Dと連携しやすくなっています。

３Ｄサウンドバッファの生成

3Dサウンドを制御するには、DirectSoundBuffer8オブジェクトとDirectSound3DBuffer8オブジェクトの２つが必要になります。DirectSound3DBuffer8オブジェクトは、3D空間上のバッファの制御を行うことができます。このオブジェクトでは、再生やボリュームの設定といったサウンドバッファの制御を行うことはできません。これらは、DirectSoundBuffer8オブジェクトで行います。
3Dサウンドバッファを生成するには、サウンドバッファの生成時にDSBCAPS_CTRL3Dフラグを指定します。このとき、パンの設定はDirectSound3Dが行うため、DSBCAPS_CTRLPANフラグは指定できません。また、guid3DAlgorithmメンバに3Dアルゴリズムを指定する必要があります。指定できるアルゴリズムは、以下のものです。なお、チャンネル数が２以上の場合、3Dサウンドバッファを生成することはできません。

DS3DALG_DEFAULT	デフォルトのアルゴリズム。現在は、DS3DALG_NO_VIRTUALIZATIONが選択されます。
DS3DALG_NO_VIRTUALIZATION	3Dサウンドをステレオパンとボリュームコントロールでエミュレートします。最も効率的なアルゴリズムですが、HRTF処理は提供されません。
DS3DALG_HRTF_LIGHT	CPUにあまり負荷をかけないHRTF(頭部伝達関数)アルゴリズムで3Dサウンドが処理されます。
DS3DALG_HRTF_FULL	高品質なHRTFアルゴリズムで3Dサウンドが処理されます。

DS3DALG_HRTF_LIGHTとDS3DALG_HRTF_FULLは、サウンドカードがWDMドライバのとき使用できます。使用できない場合は、DS3DALG_NO_VIRTUALIZATIONで代用されます。

サウンドバッファが生成できたら、DirectSoundBuffer8オブジェクトのQueryInterfaceメソッドでDirectSound3DBuffer8オブジェクトを取得します。

```
// 3Dサウンドバッファ生成
// フォーマット設定
WAVEFORMATEX   wfx;
ZeroMemory(&wfx, sizeof(wfx));
wfx.wFormatTag      = WAVE_FORMAT_PCM;
wfx.nSamplesPerSec  = 22050;
wfx.wBitsPerSample  = 8;
wfx.nChannels       = 1;
wfx.nBlockAlign     = wfx.wBitsPerSample / 8 * wfx.nChannels;
wfx.nAvgBytesPerSec = wfx.nBlockAlign * wfx.nSamplesPerSec;
wfx.cbSize          = 0;

// セカンダリバッファ能力設定
DSBUFFERDESC   dsbd;
```

```
    ZeroMemory(&dsbd, sizeof(dsbd));
    dsbd.dwSize          = sizeof(dsbd);
    dsbd.dwFlags         = DSBCAPS_LOCDEFER | DSBCAPS_CTRLVOLUME | DSBCAPS_CTRL3D |
                           DSBCAPS_GETCURRENTPOSITION2;
    dsbd.dwBufferBytes   = wfx.nAvgBytesPerSec * 4;  // ４秒
    dsbd.lpwfxFormat     = &wfx;
    dsbd.guid3DAlgorithm = DS3DALG_HRTF_LIGHT;

    // セカンダリバッファ生成(lpDSound8は、初期化済みのDirectSound8オブジェクト)
    LPDIRECTSOUNDBUFFER   lpdsb = NULL;
    lpDSound8->CreateSoundBuffer(&dsbd, &lpdsb, NULL);

    // DirectSoundBuffer8オブジェクト取得
    LPDIRECTSOUNDBUFFER8   lpDSBuffer8 = NULL;
    lpdsbd->QueryInterface(IID_IDirectSoundBuffer8, (LPVOID*)lpDSBuffer8);

    lpdsb->Release();      // DirectSoundBufferバージョン１は不要なので解放する

    // DirectSound3DBuffer8オブジェクト取得
    LPDIRECTSOUND3DBUFFER8   lpDS3DBuffer8 = NULL;    // 3Dサウンドバッファオブジェクト
    lpDSBuffer8->QueryInterface(IID_IDirectSound3DBuffer8, (LPVOID*)&lpDS3DBuffer8);
```

DirectSound3DBuffer8オブジェクトの解放は、Releaseメソッドで行います。

○ 課題 ○

3Dサウンドバッファを生成する処理を追加しましょう。

(１)3Dサウンドバッファオブジェクトの配列を適切な場所に追加しましょう。

－追加１－
```
static LPDIRECTSOUND3DBUFFER8   g_lpDS3DBuf8[DXA_SB_MAX] = {NULL}; // DirectSound3DBufferオブジェクト
```

(２)DXACreateBuffer関数とDXACreateBufferFromFile関数のプロトタイプを以下のように変更します。b3DBufferは、3Dサウンドバッファを生成するときにtrue、そうでないときにfalseを指定します。省略することも可能で、その場合はfalseになります。

－変更１－
```
bool DXACreateBuffer(const DXA_SB dxaBuf, LPWAVEFORMATEX lpwfxFormat, const DWORD dwBufferBytes,
                     const bool b3DBuffer = false);
bool DXACreateBufferFromFile(const DXA_SB dxaBuf, LPSTR lpszFileName, const bool b3DBuffer = false);
```

(３)DXACreateBuffer関数とDXACreateBufferFromFile関数の定義部分を以下のように変更します。

－変更２－
```
bool DXACreateBuffer(const DXA_SB dxaBuf, LPWAVEFORMATEX lpwfxFormat, const DWORD dwBufferBytes,
                     const bool b3DBuffer)
{
```

－変更３－
```
bool DXACreateBufferFromFile(const DXA_SB dxaBuf, LPSTR lpszFileName, const bool b3DBuffer)
{
```

(４)DXACreateBuffer関数のセカンダリバッファ能力設定の部分を以下のように変更します。

－変更４－
```
    // セカンダリバッファ能力設定
    DSBUFFERDESC   dsbd;
    ZeroMemory(&dsbd, sizeof(dsbd));
    dsbd.dwSize          = sizeof(dsbd);
    dsbd.dwBufferBytes   = dwBufferBytes;
    dsbd.dwReserved      = 0;
    dsbd.lpwfxFormat     = lpwfxFormat;
    dsbd.dwFlags         = DSBCAPS_LOCDEFER | DSBCAPS_CTRLVOLUME | DSBCAPS_CTRLFX |
                           DSBCAPS_GETCURRENTPOSITION2;
    if(false == b3DBuffer) {
        dsbd.dwFlags         |= DSBCAPS_CTRLPAN;
```

```
        dsbd.guid3DAlgorithm  = GUID_NULL;
    } else {
        dsbd.dwFlags         |= ※ここは各自考えましょう
        dsbd.guid3DAlgorithm  = DS3DALG_HRTF_LIGHT;
    }
```

(５)DXACreateBuffer関数の適切な部分に以下のプログラムを追加しましょう。

－追加２－

```
    // ３Ｄサウンドバッファ取得
    if(true == b3DBuffer) {
        if(S_OK != g_lpDSBuf8[dxaBuf]->※ここは各自考えましょう)
            OutputDebugString("*** Error - ３Ｄサウンドバッファ取得失敗(DXACreateBuffer)\n");
    }
```

(６)DXACreateBufferFromFile関数のバッファ生成の部分を以下のように変更します。

－変更５－

```
    // バッファ生成
    if(false == DXACreateBuffer(dxaBuf, &wfxDecode, dwDecodeSize, b3DBuffer)) {
        OutputDebugString("*** Error - バッファ生成失敗(DXACreateBufferFromFile)\n");
        return false;
    }
```

(７)DirectSound3DBuffer8オブジェクトを解放する処理をDXAReleaseBuffer関数に追加しましょう。