オブジェクト指向と
ゲームプログラミング

▶ DirectX Audio編 ― 第１３回　３Ｄバッファの設定 ◀

３Ｄバッファパラメータの設定

DirectSound3Dでは、3Dバッファごとに位置、速度、コーンといったパラメータの設定を行うことができます。これらのパラメータは、再生中でも停止中でも行うことができ、リアルタイムに反映されます。

位置の設定

3Dバッファの位置は、IDirectSound3DBuffer8::SetPositionメソッドで設定します。また、GetPositionメソッドで現在の位置を取得することができます。

```
// 位置の設定(pDS3DBufferは初期化済みのDirectSound3DBufferオブジェクト)
pDS3DBuffer->SetPosition(10.0f, 0.0f, -4.0f, DS3D_IMMEDIATE);    // 左からｘ，ｙ，ｚ位置
```

速度の設定

3Dバッファの速度は、IDirectSound3DBuffer8::SetVelocityメソッドで設定します。また、GetVelocityメソッドで現在の速度を取得することができます。

```
// 速度設定(左からｘ，ｙ，ｚの速度ベクトル)
pDS3DBuffer->SetVelocity(1.0f, 0.0f, 0.0f, DS3D_IMMEDIATE);
```

この速度は、DirectSound3Dがドップラー効果を計算するための値であり、実際にバッファが移動するわけではありません。

最小距離と最大距離の設定

DirectSound3Dでは、リスナが3Dバッファに近づくにつれて音量が大きくなり、距離が半分になると音量は倍になります。しかし、ある距離より近づいたときに音量が増加し続けるのは不都合となる場合があります。最小距離とは、この「ある距離」のことです。最小距離より近づいても音量は大きくならず、最小距離より離れると音量が小さくなります。
最小距離は、IDirectSound3DBuffer8::SetMinDistanceメソッドで行います。デフォルト値はDS3D_DEFAULTMINDISTANCEで、１メートルと定義されています。

```
// 最小距離を0.02メートル(2cm)に設定
pDS3DBuffer->SetMinDistance(0.02f, DS3D_IMMEDIATE);
```

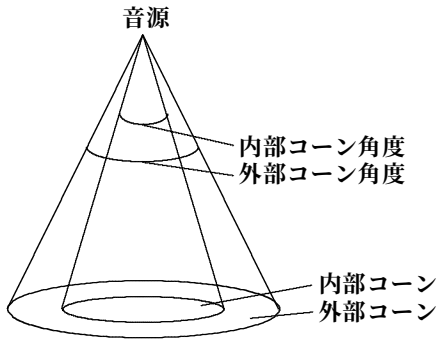
最小距離のほかに最大距離も設定できます。最大距離とは、そこを超えるとそれ以上サウンドの音量が小さくならない距離のことです。最大距離は、IDirectSound3DBuffer8::SetMaxDistanceメソッドで行います。デフォルト値はDS3D_DEFAULTMAXDISTANCEで、10億メートルと定義されています。

```
// 最大距離を100メートルに設定
pDS3DBuffer->SetMinDistance(100.0f, DS3D_IMMEDIATE);
```

DirectSound3Dでは、リスナーとサウンドバッファの距離が最小距離から最大距離の間にあるとき、音量が変化します。

サウンドコーンの設定

サウンドコーンとは、方向のあるサウンドを表すためのモデルのことです。サウンドコーンは、Direct3Dのスポットライトのような形になっており、内部コーンと外部コーンで形成されます。リスナーが内部コーンの中にいる場合、サウンドの音量はリスナとの距離や向きを考慮したものになります。内部コーンの外側かつ外部コーンの内側である場合は、設定された係数によってサウンドの音量がさらに小さくなります。コーンから完全に外れた場合は、サウンドは聞こえなくなります。



コーンの向きはIDirectSound3DBuffer8::SetConeOrientationメソッド、コーンの角度はSetConeAnglesメソッド、外部コーンの音量(の減衰量)はSetConeOutsideVolumeメソッドで行います。また、コーンのそれぞれの設定値はGetConeOrientationメソッド、GetConeAngleメソッド、GetConeOutsideVolumeメソッドで取得することができます。

```
// コーン設定
pDS3DBuffer->SetConeOrientation(0.0f, 0.0f, 1.0f, DS3D_IMMEDIATE); // 引数は方向ベクトル
pDS3DBuffer->SetConeAngles(90, 120, DS3D_IMMEDIATE);     // 左から内部、外部コーンの角度
pDS3DBuffer->SetConeOutsideVolume(-1000, DS3D_IMMEDIATE);      // 外部コーンでは10dB減衰
```

操作モードの設定

IDirectSound3DBuffer8::SetModeメソッドで３Ｄサウンド処理の操作モードを設定することができます。操作モードは以下の３つが用意されています。

DS3DMODE_DISABLE	３Ｄサウンド処理を無効にします。サウンドは、リスナの頭の中心で発生しているように聞こえます。
DS3DMODE_HEADRELATIVE	サウンドパラメータ(位置、速度、向き)は、リスナのパラメータに対して相対的なものになります。このモードでは、リスナのパラメータが変化するとサウンドの絶対パラメータは自動的に更新されるので、相対パラメータは一定に保たれます。
DS3DMODE_NORMAL	標準の処理を行います。デフォルトは、このモードです。

```
// モード設定(相対モード)
pDS3DBuffer->SetMode(DS3DMODE_HEADRELATIVE, DS3D_IMMEDIATE);
```

すべてのパラメータの設定

IDirectSound3DBuffer8::SetAllParametersメソッドは、3Dサウンドバッファに関するすべてのパラメータの設定を行うことができます。また、GetAllParametersメソッドはすべてのパラメータを取得することができます。これらのメソッドを呼び出す前に、引数で渡すDS3DBUFFER構造体のdwSizeメンバに構造体のサイズを設定しておく必要があります。

```
// すべてのパラメータの設定
DS3DBUFFER   ds3db;
ZeroMemory(&ds3dBuf, sizeof(ds3db));
ds3db.dwSize             = sizeof(ds3db);
ds3db.vPosition          = D3DXVECTOR3(1.0f, 1.0f, 0.0f);   // 位置
ds3db.vVelocity          = D3DXVECTOR3(0.0f, 0.0f, 1.0f);   // 速度ベクトル
ds3db.dwInsideConeAngle  =  90;                             // 内部コーン角度
ds3db.dwOutsideConeAngle = 120;                             // 外部コーン角度
ds3db.vConeOrientation   = D3DXVECTOR3(0.0f, 0.0f, -1.0f); // 方向ベクトル
ds3db.lConeOutsideVolume = -100;                            // 外部コーンでの減衰量
ds3db.flMinDistance      = 0.5f;                            // 最小距離
ds3db.flMaxDistance      = 1000.0f;                         // 最大距離
ds3db.dwMode             = DS3DMODE_NORMAL;                 // 操作モード
pDS3DBuffer->SetAllParameters(&ds3db, DS3D_IMMEDIATE);
```

課　題

CSound3DBufferクラスに、各パラメータを設定する機能を追加しましょう。

(１)ISound3DBufferクラスに、各パラメータを設定する純粋仮想関数を宣言します。以下のプログラムを適切な場所に追加しましょう。

```
virtual void SetPosition(const D3DVALUE x, const D3DVALUE y, const D3DVALUE z) = 0;
virtual void SetVelocity(const D3DVALUE x, const D3DVALUE y, const D3DVALUE z) = 0;

virtual void SetConeAngles(const DWORD inInside, const DWORD inOutside) = 0;
virtual void SetConeOrientation(const D3DVECTOR& inOrientation) = 0;
virtual void SetConeOutsideVolume(const LONG inVolume) = 0;
virtual void SetConeParameters(const DWORD inInside, const DWORD inOutside,
                               const D3DVECTOR& inOrientation, const LONG inVolume) = 0;

virtual void SetDistance(const D3DVALUE inMin, const D3DVALUE inMax) = 0;
virtual void SetMode(const DWORD inMode) = 0;
virtual void SetAllParameters(const DS3DBUFFER& in3DBuffer) = 0;
```

(２)各関数の仕様は、以下のとおりです。

SetPosition　設定

3Dバッファの位置を設定します。

試`void SetPosition(const D3DVALUE x, const D3DVALUE y, const D3DVALUE z);`

x	3Dバッファのｘ座標
y	3Dバッファのｙ座標
z	3Dバッファのｚ座標

SetVelocity　設定

3Dバッファの速度を設定します。

試`void SetVelocity(const D3DVALUE x, const D3DVALUE y, const D3DVALUE z);`

x	ｘ方向の速度ベクトル
y	ｙ方向の速度ベクトル
z	ｚ方向の速度ベクトル

SetConeAngles　設定

3Dバッファのコーンの角度を設定します。

試`void SetConeAngles(const DWORD inInside, const DWORD inOutside);`

inInside	内部コーンの角度
inOutside	外部コーンの角度

SetConeOrientation　設定

コーンの向きを設定します。

試`void SetConeOrientation(const D3DVECTOR& inOrientation);`

inOrientation	コーンの向きベクトル

SetConeOutsidevolume　設定

コーン外部の音量を設定します(増幅はできません)。

試`void SetConeOutsideVolume(const LONG inVolume);`

inVolume	コーン外部の音量(減衰量)

SetConeParameters　設定

コーンのすべてのパラメータを設定します。

試
```
void SetConeParameters(const DWORD inInside, const DWORD inOutside,
                       const D3DVECTOR& inOrientation, const LONG inVolume);
```

inInside	内部コーンの角度
inOutside	外部コーンの角度
inOrientation	コーンの向きベクトル
inVolume	コーン外部の音量(減衰量)

SetDistance　設定

バッファが減衰を始めるリスナからの距離(最小距離)と、それ以上減衰しなくなるリスナからの距離(最大距離)を設定します。

書式void SetDistance(const D3DVALUE inMin, const D3DVALUE inMax);

inMin　　最小距離
inMax　　最大距離

SetMode　　設定

3Dバッファの操作モードを設定します。

書式void SetMode(const DWORD inMode);

inMode　　操作モード

SetAllParameters　　設定

3Dバッファのすべてのパラメータを設定します。

書式void SetAllParameters(const DS3DBUFFER& in3DBuffer);

in3DBuffer　　３Ｄバッファのパラメータが格納されたDS3DBUFFER構造体の変数

(３)関数の仕様をもとに、CSound3DBufferクラスに各関数を実装しましょう。

```
/*****************************************************************************************************/
/*                                              位置設定                                             */
/*****************************************************************************************************/
void CSound3DBuffer::SetPosition(const D3DVALUE x, const D3DVALUE y, const D3DVALUE z)
{
    m_pDS3DBuffer->※ここは各自考えましょう;
}

/*****************************************************************************************************/
/*                                              速度設定                                             */
/*****************************************************************************************************/
void CSound3DBuffer::SetVelocity(const D3DVALUE x, const D3DVALUE y, const D3DVALUE z)
{
    m_pDS3DBuffer->※ここは各自考えましょう;
}

/*****************************************************************************************************/
/*                                          コーン角度設定                                           */
/*****************************************************************************************************/
void CSound3DBuffer::SetConeAngles(const DWORD inInside, const DWORD inOutside)
{
    m_pDS3DBuffer->※ここは各自考えましょう;
}

/*****************************************************************************************************/
/*                                          コーン向き設定                                           */
/*****************************************************************************************************/
void CSound3DBuffer::SetConeOrientation(const D3DVECTOR& inOrientation)
{
    m_pDS3DBuffer->????????????????????(inOrientation.x, inOrientation.y, inOrientation.z, DS3D_IMMEDIATE);
}

/*****************************************************************************************************/
/*                                        コーン外部音量設定                                         */
/*****************************************************************************************************/
void CSound3DBuffer::SetConeOutsideVolume(const LONG inVolume)
{
    m_pDS3DBuffer->※ここは各自考えましょう;
}

/*****************************************************************************************************/
/*                                        コーンパラメータ設定                                       */
/*****************************************************************************************************/
void CSound3DBuffer::SetConeParameters(const DWORD inInside, const DWORD inOutside,
                                       const D3DVECTOR& inOrientation, const LONG inVolume)
{
    SetConeAngles(inInside, inOutside);
    SetConeOrientation(inOrientation);
    SetConeOutsideVolume(inVolume);
}

/*****************************************************************************************************/
/*                                              距離設定                                             */
/*****************************************************************************************************/
void CSound3DBuffer::SetDistance(const D3DVALUE inMin, const D3DVALUE inMax)
```

```
{
    m_pDS3DBuffer->※ここは各自考えましょう(inMin, DS3D_IMMEDIATE);
    m_pDS3DBuffer->※ここは各自考えましょう(inMax, DS3D_IMMEDIATE);
}

/*****************************************************************************************************/
/*                                          モード設定                                               */
/*****************************************************************************************************/
void CSound3DBuffer::SetMode(const DWORD inMode)
{
    m_pDS3DBuffer->※ここは各自考えましょう;
}

/*****************************************************************************************************/
/*                                        全パラメータ設定                                           */
/*****************************************************************************************************/
void CSound3DBuffer::SetAllParameters(const DS3DBUFFER& in3DBuffer)
{
    m_pDS3DBuffer->※ここは各自考えましょう;
}
```

(４)CNullSound3DBufferクラスに各関数を「何もしない」処理で実装しましょう。

(５)CSound3DBufferクラスが正しく動作するか確認します。以下のようにプログラムを追加、変更しましょう。

・CTestSceneクラスのメンバに追加

```
ISoundBuffer*   m_pSE;
float           m_Angle;
```

・CTestSceneクラスのコンストラクタ初期化子に追加

```
m_Angle(0.0f)
```

・CTestScene::CTestScene関数に追加

```
m_pSE = DXAudio().CreateBufferFromFile("Wav\\SE.wav", true);
m_pSE->Play(0, DSBPLAY_LOOPING);
```

・CTestScene::~CTestScene関数に追加

```
DXAudio().ReleaseAllBuffers();
```

・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
    // 内部処理
    // 角度処理
    m_Angle += 1.0f;
    if(m_Angle >= 360.0f)
        m_Angle = 0.0f;

    // 座標設定
    const float   Radian = m_Angle * 3.141592654f / 180.0f;
    m_pSE->Get3DBuffer()->SetPosition(::cos(Radian), 0.0f, ::sin(Radian));

    // 描画処理
    if(FPSTimer().IsSkip() == true)
        return 0;

    return 0;
}
```