ゲームプログラミング

▶　第８回　MCIによるMIDI演奏　◀

MCI(Media Control Interface)を利用すると、AVIムービーやWAVEオーディオといったマルチメディアファイルを簡単に扱うことができます。

ＭＣＩ

MCIは、マルチメディアファイルを簡単に扱うためのインタフェースです。これを利用するためのAPIは便利なものが多く用意されており、マルチメディアファイルの再生や録音といったことが簡単にプログラムできます。しかし、関連デバイスが初期化されるまでプログラムが停止してしまうという欠点もあります。

mciSendString関数

mciSendString関数は、コマンド文字列をMCIデバイスに送出することで、マルチメディアファイルの読み込み、再生、停止といった基本的な操作を行うことができます。

```
MCIERROR mciSendString(LPCTSTR lpszCommand,          // コマンド文字列
                       LPTSTR  lpszReturnString,     // バッファへのポインタ
                       UINT    cchReturn,            // バッファのサイズ
                       HANDLE  hwndCallback)         // 通知ウィンドウのハンドル
```

１つ目の引数はMCIデバイスに送出するコマンド文字列です。２つ目と３つ目の引数はほとんどの場合使用しないので、それぞれNULLと0を指定します。４つ目の引数はウィンドウのハンドルを指定します。ここで指定したウィンドウにMCIデバイスの操作状況がメッセージとして通知されます。
戻り値は、関数が成功すれば０以外の値、それ以外の場合は０が返されます。

mciSendString関数はリンク時にスタティックライブラリ"winmm.lib"が必要になります。VisualC++のメニューからﾌﾟﾛｼﾞｪｸﾄ(P)→設定(S)→リンクで指定するか、ソースファイルでプリプロセッサ「#pragma comment(lib, ライブラリ名)」で指定してリンクさせるようにします。

```
#pragma comment(lib, "winmm.lib")
```

コマンド文字列

mciSendString関数に送出するコマンド文字列は、いくつかの英単語を組み合わせた形式になっています。

MCIデバイスの初期化

デバイスを初期化するには、コマンド文字列
　"close all"
を送出します。このコマンドは、すべてのMCIデバイスを閉じて初期状態に戻します。再生中にこのコマンドを送出すると、再生を停止し、読み込んだファイルを解放してデバイスを閉じます。このコマンド送出後に再生を行う場合は、デバイスとファイルを開く必要があります。
このコマンド文字列は、再生を完全に停止したいときに使用することもできます。

```
mciSendString("close all", NULL, 0, NULL)
```

ファイルを読み込み、MIDIデバイスの準備をする

再生するファイルを読み込み、MIDIデバイスの準備をするには、コマンド文字列
　"open *ファイル名* type sequencer alias MUSIC"
を送出します。「*ファイル名*」の部分は、MIDIファイルが格納されているパスを記述します。たとえば、Dドライブの"MIDI"フォルダにある"kitto.mid"を読み込む場合は、"D:\\MIDI\\kitto.mid"となります。'\'マークが２つ必要となることに注意しましょう。

```
mciSendString("open D:\\MIDI\\kitto.mid type sequencer alias MUSIC", NULL, 0, NULL)
```

MIDIファイルの再生

読み込んだMIDIファイルを演奏するには、コマンド文字列
　"play MUSIC from 0 notify"
を送出します。このコマンドの"from 0"はデータのどこから再生するかの指定で、０は先頭になります。

最後の"notify"は、再生終了をウィンドウに通知したい場合に使います。これを指定すると、再生終了時に"MM_MCINOTIFY"メッセージが発生するようになります。notifyコマンドを使う場合は、mciSendString関数の４つ目の引数に通知先ウィンドウのハンドルを指定する必要があります。

```
mciSendString("play MUSIC from 0 notify", NULL, 0, hWnd)       ※hWndはウィンドウのハンドル
```

(演奏の停止と再開)

再生を停止するには、コマンド文字列

"stop MUSIC"

を送出します。このコマンドは演奏を完全に停止するのではなく一時停止状態します。一時停止状態から復帰するには、コマンド文字列

"play MUSIC notify"

を送出します。再生とほとんど同じコマンドですが、"from ～"がないため停止位置から演奏を再開することができます。最後の"notify"は繰り返し演奏したい場合に必要です。

```
一時停止：mciSendString("stop MUSIC", NULL, 0, NULL)
演奏再開：mciSendString("play MUSIC notify", NULL, 0, hWnd)   ※hWndはウィンドウのハンドル
(リピートなし：mciSendString("play MUSIC", NULL, 0, NULL))
```

(繰り返し演奏)

再生が終了したときに繰り返し再生させたい場合は、再生時に"notify"コマンドを指定しておく必要があります。こうすると、再生終了時にMM_MCINOTIFYメッセージがウィンドウに通知されます。このメッセージの追加情報WPARAMには操作状況が格納されます。ここが"MCI_NOTIFY_SUCCESSFUL"の場合、再生が成功したことを表します。このとき再び再生コマンドを送出すれば、繰り返し再生できます。曲の途中から繰り返させたい場合は、"from 0"の０を適当な値に変更します。

```
case MM_MCINOTIFY:
  if(MCI_NOTIFY_SUCCESSFUL == wParam)
       mciSendString("play MUSIC from 0 notify", NULL, 0, hWnd);
  return 0;
```

○ 課題 ○

MIDIファイルの再生用関数群"MIDIUtils.cpp"とヘッダファイル"MIDIUtils.h"を完成させて標準MIDIファイル(.MID)を再生してみましょう。

MIDIUtils.hとMIDIUtils.cppはこれまで使っていたプロジェクトに追加します。

あるソースファイルでMIDIUtils.cppの関数群を使用したい場合は、そのソースファイルでMIDIUtil.hヘッダをインクルードします。たとえば、WinMain.cppというソースファイルでMIDIUtilsの関数を使用したい場合は、WinMain.cppでMIDIUtils.hをインクルードします。

(１)MIDIUtils.hの作成

ヘッダファイルには、cppファイルで定義する関数のプロトタイプや、定数、構造体、列挙などを定義します。MIDIUtils.hヘッダでは、MIDIUtils.cppで定義する関数のプロトタイプを記述します。

```
/*
================================================================================================
                                        Windowsプログラミング
    Programmed by Hibikino software. Copyright (c) 2001 Hibikino software. All rights reserved.
================================================================================================

【対象ＯＳ】
   Microsoft Windows95/NT4.0以降

【コンパイラ】
   Microsoft VisualC++ 6.0J ServicePack5

【プログラム】
   MIDIUtils.h
              MIDIファイル演奏ユーティリティヘッダ

【履歴】
   * Version    0.00       2001/11/14  19:35:32

================================================================================================
*/
```

```
/*****************************************************************************************************/
/*                                          MIDIファイル読み込み                                     */
/*****************************************************************************************************/
bool MIDILoadFromFile(LPCTSTR lpszMidiName)
{
    MIDIStop();

    // 指定されたファイル名でコマンドを編集し、ファイルを読み込む
    TCHAR   szMCIStr[300];
    wsprintf(szMCIStr, "open %s type sequencer alias MUSIC", lpszMidiName);
    if(0 != mciSendString(szMCIStr, NULL, 0, NULL))
        return ?????;   // 失敗

    return ????;        // 成功
}
```

(２)MIDIUtils.cppの作成
.cppファイルには、関数本体を定義します。

MIDILoadFromFile関数は、MIDIデバイスを開き、引数で指定された標準MIDIファイルを読み込みます。戻り値は関数が成功すればtrue、それ以外はfalseを返します。
この関数内で使用されているwsprintf関数は、１つ目の引数で指定した文字型配列に対してprintf関数を行います。ディスプレイに表示されるかわりに文字列が生成され、指定した領域に格納されるというものです。なお、wsprintf関数は浮動小数点の出力はできません(sprintf関数では可能です)。

MIDIPlay関数は、読み込まれているMIDIファイルを再生します。引数は再生終了を通知するウィンドウです。この引数がNULLの場合は繰り返しません。戻り値は関数が成功すればtrue、それ以外はfalseを返します。

```
/*****************************************************************************************************/
/*                                          再　　生                                                 */
/*****************************************************************************************************/
bool MIDIPlay(const HWND hWnd)
{
    g_hNotifyWnd = NULL;    // 再生前に通知ウィンドウを初期化

    // オープンしたファイルを演奏
    if(0 != mciSendString("※コマンド文字列は各自考えてください", NULL, 0, hWnd))
        return false;

    g_hNotifyWnd = hWnd;    // 再生に成功した場合、通知ウィンドウを保存しておく(演奏再開時に必要)

    return true;
}
```

MIDIStop関数は、再生を完全に停止し、デバイスと読み込まれているファイルを解放します。戻り値は関数が成功すればtrue、それ以外はfalseを返します。この関数の実行すると、MIDIファイルを読み込むまで再生できなくなります。

```
/*****************************************************************************************************/
/*                                          停　　止                                                 */
/*****************************************************************************************************/
bool MIDIStop()
{
    MIDIPause();    // 演奏を停止する

    // デバイスをすべて閉じる
    if(0 != mciSendString("※コマンド文字列は各自考えてください", NULL, 0, NULL))
        return false;

    return true;
}
```

MIDIRepaet関数は、MIDIデータの先頭から再生を行います。この関数は、MM_MCINOTIFYメッセージに応答するときに使用します。戻り値は関数が成功すればtrue、それ以外はfalseを返します。

```
/*****************************************************************************************************/
/*                                              リピート                                             */
/*****************************************************************************************************/
bool MIDIRepeat()
{
    if(0 != mciSendString("play MUSIC from 0 notify", NULL, 0, g_hNotifyWnd))
        return ?????;

    return ????;
}
```

MIDIPause関数は、演奏を一時停止します。戻り値は関数が成功すればtrue、それ以外はfalseを返します。この関数を実行してもデバイスや読み込まれたファイルは解放されないので、"play MUSIC"コマンドを送出すれば、停止した位置から再生を再開できます。

```
/*****************************************************************************************************/
/*                                              一時停止                                             */
/*****************************************************************************************************/
bool MIDIPause()
{
    if(0 != mciSendString("※コマンド文字列は各自考えてください", NULL, 0, NULL))
        return false;

    return true;
}
```

MIDIResume関数は、停止位置から再生を再開します。戻り値は関数が成功すればtrue、それ以外はfalseを返します。この関数は、MIDIPause関数で一時停止した後、演奏を再開したい場合に呼び出します。

```
/*****************************************************************************************************/
/*                                              再　開                                               */
/*****************************************************************************************************/
bool MIDIResume()
{
    if(0 != mciSendString("play MUSIC notify", NULL, 0, g_hNotifyWnd))
        return false;

    return true;
}
```

－MIDIUtils使用例－

・MIDIファイルを開く　　MIDILoadFromFile("D:\\MUSIC\\MIDI\\CANARY.MID");
・再生開始　　MIDIPlay(hWnd);　　// hWndは演奏終了を通知するウィンドウのハンドル
・一時停止　　MIDIPause();
・再生再開　　MIDIResume();

・繰り返し

```
    case MM_MCINOTIFY:
      if(MCI_NOTIFY_SUCCESSFUL == wParam)
          MIDIRepeat();
      return 0;
```

・演奏停止(WM_CLOSEメッセージで完全停止する場合)

```
    case WM_CLOSE:
      MIDIStop();
      DestroyWindow(hWnd);
      return 0;
```

MIDIUtils.cpp

```
/*
================================================================================================
                                    ゲームプログラミング
    Programmed by Hibikino software. Copyright (c) 2001 Hibikino software. All rights reserved.
================================================================================================

  【対象ＯＳ】
     Microsoft Windows95/NT4.0以降

  【コンパイラ】
     Microsoft VisualC++ 6.0J ServicePack5

  【プログラム】
     MIDIUtils.h
                MIDIファイル演奏ユーティリティ

  【履歴】
     * Version     0.00       2001/11/14  19:35:32
================================================================================================
*/

/************************************************************************************************/
/*                                      インクルードファイル                                    */
/************************************************************************************************/
#include "MIDIUtils.h"

/************************************************************************************************/
/*                                      スタティックライブラリ                                  */
/************************************************************************************************/
#pragma comment(lib, "winmm.lib")

/************************************************************************************************/
/*                                      グローバル変数                                          */
/************************************************************************************************/
namespace {
    HWND    g_hNotifyWnd = NULL;      // 通知ウィンドウ
}

/************************************************************************************************/
/*                                      MIDIファイル読み込み                                    */
/************************************************************************************************/


/************************************************************************************************/
/*                                      再　　生                                                */
/************************************************************************************************/


/************************************************************************************************/
/*                                      停　　止                                                */
/************************************************************************************************/


/************************************************************************************************/
/*                                      リピート                                                */
/************************************************************************************************/


/************************************************************************************************/
/*                                      一時停止                                                */
/************************************************************************************************/


/************************************************************************************************/
/*                                      再　　開                                                */
/************************************************************************************************/
```