ゲームプログラミング

▶　DirectInput－第３回　デバイスオブジェクトの生成　◀

DirectInputでは、入力デバイスごとにDirectInputDevice8オブジェクトを生成する必要があります。どのような入力デバイスでも、ほとんど同じ手順で生成し、初期化することができます。

DirectInputDevice8オブジェクトの生成と初期化

デバイスオブジェクトは、入力デバイスを管理するオブジェクトです。入力デバイスから情報を取得するメソッドなどが提供されています。このオブジェクトは以下の手順で初期化します。

①デバイスオブジェクトの生成
②データフォーマットの設定
③協調レベルの設定
④プロパティの設定
⑤アクセス権の取得

①デバイスオブジェクトの生成
デバイスオブジェクトは、DirectInputオブジェクトのCreateDeviceメソッドで生成します。

CreateDeviceメソッド
－説明－
CreateDeviceメソッドは、指定された入力デバイスのGUIDに基づきデバイス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、入力デバイスのGUID(デバイスごとに個別に割り当てられた128ビットの値)です。この値はEnumDevicesメソッドでデバイスを列挙することにより取得できます。なお、キーボードとマウスはあらかじめ定義された以下の値を指定できます。

GUID_SysKeyboard	デフォルトのキーボード
GUID_SysMouse	デフォルトのマウス

２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(LPDIRECTINPUTDEVICE8型)のアドレスです。

３つ目の引数は、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DI_OK、それ以外はエラーの原因をエラーコードで返します。

```
// デバイスオブジェクト生成(キーボード)
LPDIRECTINPUT8         lpDInput8;          // DirectInput8オブジェクト(初期化済みとする)
LPDIRECTINPUTDEVICE8   lpDIDevice8 = NULL;   // デバイスオブジェクト
lpDInput8->CreateDevice(GUID_SysKeyboard, &lpDIDevice8, NULL);
```

②データフォーマットの設定
デバイスオブジェクトが生成できたら、データフォーマットの設定を行います。データフォーマットとは、入力デバイスからの情報をどのような形式で受け取るかを定義するもので、デバイスオブジェクトのSetDataFormatメソッドで設定します。
SetDataFormatメソッドの引数は、データ形式を設定したDIDATAFORMAT構造体型変数のアドレスです。この構造体を設定することにより、アプリケーションにあわせた情報を入力デバイスから取得することができます。キーボード、マウス、ジョイスティックに関しては、あらかじめ定義されている以下の変数を指定することができます。

変　数	対応デバイス	GetDeviceStateメソッドで返される情報
c_dfDIKeyboard	キーボード	256バイトの配列
c_dfDIMouse	マウス	DIMOUSESTATE構造体
c_dfDIMouse2	マウス	DIMOUSESTATE2構造体
c_dfDIJoystick	ジョイスティック	DIJOYSTATE構造体
c_dfDIJoystick2	ジョイスティック(フォースフィードバック)	DIJOYSTATE2構造体

設定したデータ形式は、GetDeviceStateメソッドで入力デバイスの情報を取得したときに返されます。

```
// データフォーマットの設定(キーボード)
lpDIDevice8->SetDataFormat(&c_dfDIKeyboard);
```

③協調レベルの設定

DirectInputでは、入力デバイスごとに協調レベルを設定する必要があります。協調レベルとは、ほかのアプリケーションとどのように協調して使用するかを示すものです。

協調レベルの設定は、デバイスオブジェクトのSetCooperativeLevelメソッドで行います。このメソッドの１つ目の引数はメインウィンドウのハンドル、２つ目の引数は協調レベルを表すフラグで、以下の２つのモードを組み合わせて指定します。

①「DISCL_FOREGROUND(フォアグランド)」または「DISCL_BACKGROUND(バックグランド)」
フォアグランドモードでは、アプリケーションが非アクティブ状態になると自動的にデバイスの入力を停止します。バックグランドモードでは常に入力を続けます。

②「DISCL_EXCLUSIVE(排他占有)」または「DISCL_NONEXCLUSIVE(非排他)」
排他モードでは、入力デバイスのアクセス権を解放しない限り、ほかのアプリケーションがその入力デバイスを制御することができなくなります。非排他モードでは、ほかのアプリケーションの妨げになることはありません。
排他占有すると、アクセス権を解放するまでDirectInputでしか制御できなくなり、協調レベルを設定した以外のウィンドウはAPIを使っても入力を受け取ることができなくなります。キーボードやマウスを排他占有すると、それらに関するメッセージが発生しなくなり、メッセージボックスのOKボタンをクリックできなくなったりします。最悪の場合はOSを終了できなくなります。ほとんどの入力デバイスは非排他モードで十分ですが、ゲームパッドなどのフォースフィードバック機能を制御するときは、排他占有モードに設定します。

```
// 協調レベル設定(フォアグランド・非排他モードに設定。hWndはメインウィンドウのハンドル)
lpDIDevice8->SetCooperativeLevel(hWnd, DISCL_FOREGROUND | DISCL_NONEXCLUSIVE);
```

④プロパティの設定

デバイスバッファの設定や、マウス、ジョイスティックの軸モードといった入力デバイスの動作属性(プロパティ)を、SetPropertyメソッドで設定することができます。

⑤アクセス権の取得

デバイスオブジェクトの設定がすべて正しく終えたら、Acquireメソッドを呼び出しアクセス権を取得します。アクセス権を取得すると、入力デバイスから情報を取得できるようになります。

```
// アクセス権取得
lpDIDevice8->Acquire();
```

デバイスオブジェクトの解放

デバイス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が、その前にUnacquireメソッドで入力デバイスのアクセス権を解放しておきます。

```
LPDIRECTINPUTDEVICE8   lpDIDevice8;      // 解放するデバイスオブジェクト(初期化済みとする)
lpDIDevice8->Unacquire();                // アクセス権解放
lpDIDevice8->Release();                  // デバイスオブジェクト解放
```

○ 課題 ○

キーボードとマウスのデバイスオブジェクトの初期化と解放処理を追加しましょう。

(１)DIUtilsでは、デバイスオブジェクトのインタフェースを配列にまとめて扱います。配列で管理するのは、デバイスごとに変数を与える方法、たとえば以下のように

```
LPDIRECTINPUTDEVICE8   lpDIDKeyboard;      // キーボード
LPDIRECTINPUTDEVICE8   lpDIDMouse;         // マウス
LPDIRECTINPUTDEVICE8   lpDIDGamepad;       // ゲームパッド
```

とするよりも利点が多いためです。たとえば、すべてのデバイス対して同じ操作を行いたい場合、配列ならループを回すだけで処理することができます。しかし、たとえば

```
    LPDIRECTINPUTDEVICE8   lpDiDevice8[4];
```

と宣言した場合、lpDIDevice8[0]はキーボードなのかそれともマウスなのかソースをよく解析しないとわかりにくくなります。そこで列挙体を利用します。
DIUtils.hに入力デバイスを指定するための列挙体を定義します。これを添字として使用します。

－追加１－

```
// デバイス指定
enum DIDEV {
    DID_KEYBOARD,      // キーボード
    DID_MOUSE,         // マウス
    DID_GAMEPAD1,      // ゲームパッド１
    DID_GAMEPAD2,      // ゲームパッド２
    DID_MAX            // デバイス最大数
};
```

上記の定義に過不足がある場合は、各自調整してください。次に、DIUtils.cppのグローバル変数として、デバイスオブジェクトのインタフェースを配列で定義します。

－追加２－

```
static LPDIRECTINPUTDEVICE8    g_lpDIDevice8[DID_MAX] = {NULL};    // DirectInputDevice8オブジェクト
```

(２)以下のプログラムは、キーボードを管理するデバイスオブジェクトを初期化するDIInitKeyboard関数です。関数の仕様をよく読み、プログラムを完成させましょう。なお、この関数は(４)で作成する関数も必要です。

DIInitKeyboard関数
－説明－
DIInitKeyboard関数は、キーボードを管理するデバイスオブジェクトを生成し、キーボードから入力を受け取るための初期化処理を行います。

－パラメータ－
const HWND hWnd…メインウィンドウのハンドル。協調レベルの設定に必要

－戻り値－
初期化に成功した場合はtrue、それ以外はfalseを返す。

－追加３－

```
/*****************************************************************************************************/
/*                                          キーボード初期化                                           */
/*****************************************************************************************************/
bool DIInitKeyboard(const HWND hWnd)
{
#ifdef _DEBUG
    if(NULL == g_lpDInput8) {
        OutputDebugString("*** Error - DirectInput未初期化(DIInitKeyboard)\n");
        return false;
    }
#endif
    DIReleaseDevice(DID_KEYBOARD);

    // デバイスオブジェクト生成
    if(DI_OK != g_lpDInput8->???????????(???????????????, &g_lpDIDevice8[DID_KEYBOARD], NULL))
        return false;

    // データフォーマット設定
    if(DI_OK != ※ここは各自考えましょう) {
        OutputDebugString("*** Error - データフォーマット設定失敗(DIInitKeyboard)\n");
        DIReleaseDevice(DID_KEYBOARD);
        return false;
    }

    // 協調レベル設定
    if(DI_OK != ※ここは各自考えましょう[フォアグランド・非排他モードに設定しましょう]) {
        OutputDebugString("*** Error - 協調レベル設定失敗(DIInitKeyboard)\n");
        DIReleaseDevice(DID_KEYBOARD);
        return false;
    }
```

```
    // アクセス権取得
    g_lpDIDevice8[DID_KEYBOARD]->※ここは各自考えましょう;

    return true;
}
```

(３)以下のプログラムは、マウスを管理するデバイスオブジェクトを初期化するDIInitMouse関数です。関数の仕様をよく読み、プログラムを完成させましょう。なお、この関数は(４)で作成する関数も必要です。

DIInitMouse関数
－説明－
DIInitMouse関数は、マウスを管理するデバイスオブジェクトを生成し、マウスから入力を受け取るための初期化処理を行います。

－パラメータ－
const HWND hWnd…メインウィンドウのハンドル。協調レベルの設定に必要

－戻り値－
初期化に成功した場合はtrue、それ以外はfalseを返す。

－追加４－

```
/*****************************************************************************************************/
/*                                          マウス初期化                                             */
/*****************************************************************************************************/
bool DIInitMouse(const HWND hWnd)
{
#ifdef _DEBUG
    if(NULL == g_lpDInput8) {
        OutputDebugString("*** Error - DirectInput未初期化(DIInitMouse)\n");
        return false;
    }
#endif
    DIReleaseDevice(DID_MOUSE);

    // デバイスオブジェクト生成
    if(DI_OK != ※ここは各自考えましょう)
        return false;

    // データフォーマット設定
    if(DI_OK != ※ここは各自考えましょう) {
        OutputDebugString("*** Error - データフォーマット設定失敗(DIInitMouse)\n");
        DIReleaseDevice(DID_MOUSE);
        return false;
    }

    // 協調レベル設定
    if(DI_OK != ※ここは各自考えましょう[APIのGetCursorPos関数でカーソル座標を取得できるように、
                                        フォアグランド・非排他モードに設定しましょう]) {
        OutputDebugString("*** Error - 協調レベル設定失敗(DIInitMouse)\n");
        DIReleaseDevice(DID_MOUSE);
        return false;
    }

    // アクセス権取得
    ※ここは各自考えましょう;

    return true;
}
```

(４)デバイスオブジェクトを解放するDIReleaseDevice関数の仕様を参考に、関数を完成させましょう。

DIReleaseDevice関数
－説明－
DIReleaseDevice関数は、指定されたデバイスオブジェクトを解放します。

－パラメータ－
const DIDEV did…解放するデバイスの指定。DIDEV列挙体型を指定

－戻り値－
なし

－追加５－
```
/*****************************************************************************************************/
/*                                        デバイスオブジェクト解放                                     */
/*****************************************************************************************************/
void DIReleaseDevice(const DIDEV did)
{
    if(NULL == g_lpDIDevice8[did])
        return;

    g_lpDIDevice8[did]->※ここは各自考えましょう;
    g_lpDIDevice8[did]->※ここは各自考えましょう;
    g_lpDIDevice8[did] = NULL;
}
```

(５)－追加６－のたりない部分を補って適切な場所に追加し、DirectInput解放時にすべてのデバイスオブジェクトが正しく解放されるようにしましょう。

－追加６－
```
    // デバイスオブジェクト解放
    for(int did = 0; did < DID_MAX; did++)
        ?????????????((DIDEV)did);
```

(６)キーボードとマウスのデバイスオブジェクトの初期化を行うように(これらを行う関数が呼び出されるように)、あるソースファイルに変更を加えます。以下の－追加７－と－追加８－を適切な場所に追加しましょう。

－追加７－
```
    // キーボード初期化
    if(false == DIInitKeyboard(hWnd)) {
        DIRelease();
        return false;
    }
```

－追加８－
```
    // マウス初期化
    if(false == DIInitMouse(hWnd)) {
        DIRelease();
        return false;
    }
```