ゲームプログラミング

▶　DirectInput－第６回　ゲームパッドの初期化　◀

ゲームパッドは、キーボードやマウスと同じ手順で初期化することができます。しかし、ゲームパッドはそれらに比べ多機能なため、いくつかの設定作業が必要になります。

ゲームパッドの初期化

ゲームパッドの初期化手順は、以下のようになります。

①デバイスの列挙
②デバイスオブジェクトの生成
③データフォーマットの設定
④協調レベルの設定
⑤バッファサイズの設定
⑥軸の列挙
⑦軸の値の範囲設定
⑧デッドゾーンの設定
⑨アクセス権の取得

①デバイスの列挙
ゲームパッドは、DirectInputオブジェクトのEnumDevicesメソッドを呼び出し、デバイスを列挙する必要があります。キーボードやマウスを複数使用するときも、このメソッドで列挙します。

EnumDevicesメソッド
－説明－
EnumDevicesメソッドは、指定された条件を満たすデバイスを列挙します。

－パラメータ－
１つ目の引数は、列挙するデバイスのタイプです。おもに以下の値を使用します。

DI8DEVCLASS_GAMECTRL	ゲームコントローラ
DI8DEVCLASS_KEYBOARD	キーボード
DI8DEVCLASS_POINTER	マウスやトラックボールのようなポインタデバイス
DI8DEVCLASS_DEVICE	以上のいずれにも該当しないデバイス
DI8DEVCLASS_ALL	すべてのデバイス

２つ目の引数は、条件を満たすデバイスが見つかるたびに呼び出されるコールバック関数のアドレスです。

３つ目の引数は、コールバック関数に渡すLPVOID型の値です。コールバック関数が呼び出されるたびに、ここで指定した値が渡されます。

４つ目の引数は、列挙の範囲を指定するフラグです。おもに以下のフラグを指定します。

DIEDFL_ALLDEVICES	すべてのインストール済みデバイス
DIEDFL_ATTACHEDONLY	インストール済みで正しく接続されているデバイスのみ
DIEDFL_FORCEFEEDBACK	フォースフィードバック対応デバイスのみ

－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。

```
// デバイスの列挙
LPDIRECTINPUT8   lpDInput8;     // DirectInput8オブジェクト(初期化済みとする)
lpDInput8->EnumDevices(DI8DEVCLASS_GAMECTRL, (LPDIENUMDEVICESCALLBACK)DIEnumGamePadProc,
                       NULL, DIEDFL_ATTACHEDONLY);
```

２つ目の引数で指定したコールバック関数は、１つ目と４つ目の引数で指定される条件を満たすデバイスの数と同じ回数呼び出されます。
このコールバック関数はプログラマが定義する関数です。この関数が呼び出されると、１つ目の引数にデバイスの情報を格納したDIDEVICEINSTANCE構造体のアドレス、２つ目の引数にEnumDevicesメソッドで指定した３つ目の引数と同じ値が渡されます。戻り値は、列挙を継続する場合はDIENUM_CONTINUE、列挙を停止する場合はDIENUM_STOPを返すようにします。

②デバイスオブジェクトの生成
　ゲームパッドの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。このメソッドの１つ目の引数は、ゲームパッドの列挙時に取得されるデバイスのGUID(DIDEVICEINSTANCE構造体のguidInstanceメンバ)を指定します。

```
// デバイスオブジェクト生成
LPDIRECTINPUTDEVICE8   lpDIDGamePad = NULL;  // ゲームパッドのデバイスオブジェクト

// ゲームパッド列挙コールバック関数
BOOL CALLBACK DIEnumGamePadProc(LPDIDEVICEINSTANCE lpDIDInst, LPVOID lpRef)
{
    // デバイスオブジェクト生成
    if(DI_OK == lpDInput8->CreateDevice(lpDIDInst->guidInstance, &lpDIDGamePad, NULL))
        return DIENUM_STOP;    // 列挙終了

    return DIENUM_CONTINUE;    // 列挙継続
}
```

③データフォーマットの設定
　データフォーマットの設定は、SetDataFormatメソッドで行います。ゲームパッドはあらかじめ定義されている変数を指定することができます。取得したい情報にあわせて、c_dfDIJoystick(一般的なゲームパッド)またはc_dfDIJoystick2(フォースフィードバック対応)のどちらかを指定します。

```
    // データフォーマット設定
    lpDIDGamePad->SetDataFormat(&c_dfDIJoystick);
```

④協調レベルの設定
　協調レベルの設定は、SetCooperativeLevelメソッドで行います。ゲームパッドは、基本的にフォアグランド、排他占有モードに設定します。フォースフィードバックに対応させる場合は、必ず排他占有モードを指定する必要があります。

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

⑤バッファサイズの設定
　バッファサイズの設定は、キーボードやマウスとまったく同じ方法で設定できます。

⑥軸の列挙
　ゲームパッドに存在する軸に対して、報告させる値の範囲やデッドゾーン(軸が中心にあるとみなされる範囲)の設定をします。ゲームパッドによって軸の数や性質が異なるため、EnumObjectsメソッドで列挙します。

EnumObjectsメソッド
－説明－
　EnumObjectsメソッドは、デバイス上で使用可能なオブジェクトを列挙します。

－パラメータ－
　１つ目の引数は、指定したオブジェクトが見つかるたびに呼び出されるコールバック関数のアドレスです。

　２つ目の引数は、コールバック関数に渡すLPVOID型の値です。コールバック関数が呼び出されるたびに、ここで指定した値が渡されます。

　３つ目の引数は、列挙するオブジェクトのタイプを指定するフラグです。おもに以下のフラグを指定します。

DIDFT_ABSAXIS	絶対軸
DIDFT_RELAXIS	相対軸
DIDFT_AXIS	すべての軸
DIDFT_PSHBUTTON	プッシュボタン
DIDFT_TGLBUTTON	トグルボタン
DIDFT_BUTTON	すべてのボタン
DIDFT_POV	視点コントローラ
DIDFT_ALL	すべてのオブジェクト

－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。

```
    // すべての軸を列挙
    lpDIDGamePad->EnumObjects((LPDIENUMDEVICEOBJECTSCALLBACK)DIEnumGamePadAxesProc, NULL, DIDFT_AXIS);
```

１つ目の引数で指定したコールバック関数は、３つ目の引数で指定したオブジェクトの数と同じ回数呼び出されます。

このコールバック関数はプログラマが定義する関数です。この関数が呼び出されると、１つ目の引数にオブジェクトの情報を格納したPDIDEVICEOBJECTINSTANCE構造体のアドレス、２つ目の引数にEnumDevicesメソッドで指定した３つ目の引数と同じ値が渡されます。戻り値は、列挙を継続する場合はDIENUM_CONTINUE、列挙を停止する場合はDIENUM_STOPを返すようにします。

⑦軸の値の範囲の設定

すべての軸に対し、報告可能な値の範囲を設定します。ここで設定した値は、軸をもっとも傾けたときの値になります。ｘ軸の場合は、もっとも左に傾けたときに最小値、もっとも右に傾けたときに最大値となります。ｙ軸の場合は、もっとも上(前)に傾けたときに最小値、もっとも下(手前)に傾けたときに最大値となります。どの軸も中心位置の値は、最大値と最小値を足して２で割った値です。

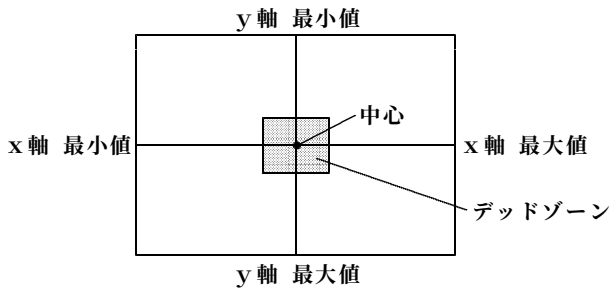
軸の値の範囲の設定は、SetPropertyメソッドで行います。１つ目の引数にDIPROP_RANGE、２つ目の引数に値の範囲を格納したDIPROPRANGE構造体に含まれるDIPROPHEADER構造体のアドレスを指定します。この構造体は、DIPROPHEADER構造体型のdiphメンバと、軸の最小値を格納するlMinメンバ、軸の最大値を格納するlMaxメンバで構成されています。diphメンバは以下のように初期化します。

メンバ	設定する値
dwSize	構造体全体のサイズ。sizeof(DIPROPDWORD)
dwHeaderSize	DIPROPHEADER構造体のサイズ。sizeof(DIPROPHEADER)
dwObj	軸を列挙した場合は、DIDEVICEOBJECTINSTANCE構造体のdwTypeメンバ
dwHow	軸を列挙した場合は、DIPH_BYID

⑧デッドゾーンの設定

軸がアナログの場合はデッドゾーンを設定します。デッドゾーンとは、軸を傾けても傾けたとはみなさない範囲のことです。デフォルト状態でのアナログの軸は、中心にあるとみなされる範囲がかなり狭くなっています。デッドゾーンを設定しないと、軸が中心にあるつもりでも傾いていると見なされ、キャラクターなどが勝手に移動してしまうことがあります。

デッドゾーンの設定は、SetPropertyメソッドで行います。１つ目の引数にDIPROP_DEADZONE、２つ目の引数に値の範囲を格納したDIPROPDWORD構造体に含まれるDIPROPHEADER構造体のアドレスを指定します。この構造体は、DIPROPHEADER構造体型のdiphメンバと、デッドゾーンの範囲を格納するdwDataメンバで構成されています。diphメンバの設定は、軸の値の範囲を設定する場合とまったく同じです。dwDataメンバは、軸を中心としたデッドゾーンの範囲を０～10,000の範囲で指定します。たとえば０ならデッドゾーンなし、5,000なら軸を中心とした50%の範囲がデッドゾーンとなります。



▲軸の範囲とデッドゾーンの関係

```
// ゲームパッド軸列挙コールバック関数
BOOL CALLBACK DIEnumGamePadAxesProc(LPCDIDEVICEOBJECTINSTANCE lpDIDObjInst, LPVOID lpRef)
{
    // 軸範囲設定
    DIPROPRANGE   dipr;
    dipr.diph.dwSize       = sizeof(dipr);
    dipr.diph.dwHeaderSize = sizeof(dipr.diph);
    dipr.diph.dwHow        = DIPH_BYID;
    dipr.diph.dwObj        = lpDIDObjInst->dwType;
    dipr.lMin              = -32767;    // 最小値
    dipr.lMax              =  32767;    // 最大値
    lpDIDGamePad->SetProperty(DIPROP_RANGE, &dipr.diph);

    // デッドゾーン設定
    DIPROPDWORD   dipdw;
    dipdw.diph.dwSize       = sizeof(dipdw);
    dipdw.diph.dwHeaderSize = sizeof(dipdw.diph);
```

```
    dipdw.diph.dwHow        = DIPH_BYID;
    dipdw.diph.dwObj        = lpDIDObjInst->dwType;
    dipdw.dwData            = 3000;     // デッドゾーンの範囲
    lpDIDGamePad->SetProperty(DIPROP_DEADZONE, &dipdw.diph);

    return DIENUM_CONTINUE;
}
```

⑨アクセス権の取得

最後に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、ゲームパッドから情報を取得できるようになります。

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

○ 課題 ○

ゲームパッドを初期化する処理を追加しましょう。

(１)DIUtils.hに、ゲームパッドの軸の範囲を定数として定義します。

－追加１－

```
enum {
    DIAXIS_RANGE_MIN    =     0,
    DIAXIS_RANGE_MAX    = 65536,
    DIAXIS_RANGE_CENTER = (DIAXIS_RANGE_MAX + DIAXIS_RANGE_MIN + 1) / 2
};
```

それぞれの定数の意味は、以下のようになります。

DIAXIS_RANGE_MIN	**軸の最小値**
DIAXIS_RANGE_MAX	**軸の最大値**
DIAXIS_RANGE_CENTER	**軸の中心値**

(２)以下のプログラムは、ゲームパッドの初期化を行うための関数群です。プログラムを解析し、完成させましょう。

```
/*****************************************************************************************************/
/*                                        ゲームパッド初期化                                           */
/*****************************************************************************************************/
bool DIInitGamePad(const HWND hWnd)
{
#ifdef _DEBUG
    if(NULL == g_lpDInput8) {
        OutputDebugString("*** Error - DirectInput未初期化(DIInitGamePad)\n");
        return false;
    }
#endif
    DIReleaseDevice(DID_GAMEPAD1);

    // ゲームパッド列挙
    g_lpDInput8->???????????(※ここは各自考えましょう, (LPDIENUMDEVICESCALLBACK)???????????????,
                             &hWnd, ※ここは各自考えましょう);
    if(NULL == g_lpDIDevice8[DID_GAMEPAD1]) {
        OutputDebugString("*** Error - ゲームパッドオブジェクト生成失敗(DICreateGamePad)\n");
        return false;
    }

    // アクセス権取得
    g_lpDIDevice8[DID_GAMEPAD1]->※ここは各自考えましょう;

    return true;
}

/*****************************************************************************************************/
/*                                   ゲームパッド列挙コールバック関数                                  */
/*****************************************************************************************************/
```

```
BOOL CALLBACK DIEnumGamePadProc(LPDIDEVICEINSTANCE lpDIDInst, LPVOID lpRef)
{
    // ゲームパッドオブジェクト生成
    if(DI_OK == g_lpDInput8->????????????(lpDIDInst->guidInstance, &g_lpDIDevice8[DID_GAMEPAD1], NULL)) {
        OutputDebugString("*** Error - ゲームパッドオブジェクト生成失敗(DIEnumGamePadProc)\n");
        return DIENUM_CONTINUE;     // 次のゲームパッドへ
    }

    // 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(DIEnumGamePadProc)\n");
        DIReleaseDevice(DID_GAMEPAD1);
        return DIENUM_CONTINUE;
    }

    // 協調レベル設定
    if(DI_OK != ※ここは各自考えましょう(*(HWND*)lRef, ※ここは各自考えましょう)) {
        OutputDebugString("*** Error - 協調レベル設定失敗(DIEnumGamePadProc)\n");
        DIReleaseDevice(DID_GAMEPAD1);
        return DIENUM_CONTINUE;
    }

    // バッファサイズ設定
    DIPROPDWORD   dipd;
    ZeroMemory(&dipd, sizeof(dipd));
    ※ここは各自考えましょう;
    ※ここは各自考えましょう;
    ※ここは各自考えましょう;
    ※ここは各自考えましょう;
    ※ここは各自考えましょう;
    if(FAILED(※ここは各自考えましょう)) {
        OutputDebugString("*** Error - バッファサイズ設定失敗(DIEnumGamePadProc)\n");
        DIReleaseDevice(DID_GAMEPAD1);
        return DIENUM_CONTINUE;
    }

    // 軸の列挙
    if(FAILED(g_lpDIDevice8[DID_GAMEPAD1]->
              ??????????((LPDIENUMDEVICEOBJECTSCALLBACK)??????????????????????, NULL, DIDFT_AXIS))) {
        DIReleaseDevice(DID_GAMEPAD1);
        return DIENUM_CONTINUE;
    }

    return DIENUM_STOP;     // 列挙終了
}

/*****************************************************************************************************/
/*                                ゲームパッド軸列挙コールバック関数                                 */
/*****************************************************************************************************/
BOOL CALLBACK DIEnumGamePadAxesProc(LPCDIDEVICEOBJECTINSTANCE lpDIDObjInst, LPVOID lpRef)
{
    // 絶対軸の場合、軸の範囲を設定
    if(0 != (lpDIDObjInst->dwType & DIDFT_ABSAXIS))
        DIPROPRANGE   dipr;
        dipr.diph.dwSize       = sizeof(dipr);
        dipr.diph.dwHeaderSize = sizeof(dipr.diph);
        dipr.diph.dwHow        = DIPH_BYID;
        dipr.diph.dwObj        = lpDIDObjInst->dwType;
        dipr.lMin              = ※ここは各自考えましょう;
        dipr.lMax              = ※ここは各自考えましょう;
        if(FAILED(g_lpDIDevice8[DID_GAMEPAD1]->※ここは各自考えましょう))
            OutputDebugString("*** Error - 軸範囲設定失敗(DISetAxesProp)\n");
    }

    // デッドゾーン設定
    DIPROPDWORD   dipdw;
    dipdw.diph.dwSize       = ※ここは各自考えましょう;
    dipdw.diph.dwHeaderSize = ※ここは各自考えましょう;
    dipdw.diph.dwHow        = ※ここは各自考えましょう;
    dipdw.diph.dwObj        = ※ここは各自考えましょう;
    dipdw.dwData            = ※ここは各自考えましょう;
    if(FAILED(g_lpDIDevice8[DID_GAMEPAD1]->※ここは各自考えましょう))
        OutputDebugString("*** Error - デッドゾーン設定失敗(DISetAxesProp)\n");
```

```
    return DIENUM_CONTINUE;
}
```

(３)ゲームパッドの初期化を行う(関数が呼び出される)ように、プログラムを変更しましょう。

(４)(２)のプログラムは、ゲームパッドが複数接続されている場合でも最初に列挙されるデバイスしか初期化しないようになっています。(２)のプログラムを変更し、ゲームパッドが複数接続されている場合、２つまで初期化するようにしましょう。