オブジェクト指向と
ゲームプログラミング

▷　フレームワーク編 － 第６回　ウィンドウとフルスクリーンの切替　◁

課　　題

たいていのゲームでは、ウィンドウモードとフルスクリーンモードというモードがあり、いつでも切り替えることができるようになっています。
ウィンドウモードとは、通常のアプリケーションのように、ウィンドウの中だけで描画を行うものです。これに対し、フルスクリーンモードとは、画面全体を占有し、ほかのアプリケーションには一切描画をさせないというものです。

Alt＋Enterキーを押したら、ウィンドウモードとフルスクリーンモードを切り替えられるようにしましょう。

(１)アプリケーションがどのモードで動作しているかを示す変数を作成します。以下のプログラムを適切な場所に追加しましょう。

```
bool    m_Windowed;  // ウィンドウモードフラグ(true…ウィンドウモード／false…フルスクリーンモード)
```

この変数は、ウィンドウモードかどうかを示すフラグです。フラグがtrueならウィンドウモード、falseならフルスクリーンモードで動作していることを示します。

(２)(１)を追加したクラスのコンストラクタ初期化子に、以下のプログラムを追加し、オブジェクト構築時にm_Windowedが適切に初期化されるようにしましょう。

```
m_Windowed(false)
```

(３)ウィンドウをモードに合わせたものに調整する関数を作成します。

ウィンドの形状、サイズ、位置をモードに合わせて調整するAdjust関数を作成します。Adjust関数はウィンドウモードなら一般的なアプリケーションのようなウィンドウを、フルスクリーンモードなら画面全体を覆う枠のないウィンドウに調整します。ウィンドウモードかどうかは引数で指定します。

以下のプログラムの足りない部分を補い、適切な場所に追加しましょう。なお、ウィンドウは画面中央に表示されるようにしましょう。

```
/*****************************************************************************************************/
/*                                          ウィンドウ調整                                           */
/*****************************************************************************************************/
bool ?????????????::Adjust(const bool inWindowed)
{
#ifdef _DEBUG
    if(m_hWnd == NULL) {
        ::OutputDebugString("*** Error - ウィンドウ未初期化(CGameWndFrame_Adjust)\n");
        return false;
    }
#endif

    // ディスプレイサイズ取得
    const int   SCREEN_WIDTH  = ::GetSystemMetrics(SM_????????);
    const int   SCREEN_HEIGHT = ::GetSystemMetrics(SM_????????);

    DWORD   style_ex = 0, style = 0;
    RECT    wnd_rect;

    if(※ここは各自考えましょう) {
        // ウィンドウモード
        style = WS_POPUPWINDOW | WS_CAPTION | WS_MINIMIZEBOX;
        ::SetRect(&wnd_rect, 0, 0, 640, 480);

        // マウスカーソル表示
        while(::??????????(TRUE) < 0)
            ;
```

```
    } else {
        // フルスクリーンモード
#ifdef NDEBUG
        style_ex = WS_EX_TOPMOST;
#endif
        style    = WS_POPUP | WS_SYSMENU;
        ::SetRect(&wnd_rect, 0, 0, SCREEN_WIDTH, SCREEN_HEIGHT);

        // マウスカーソル消去
        while(::??????????(FALSE) >= 0)
            ;
    }

    // ウィンドウスタイル設定
    ::SetWindowLong(m_hWnd, GWL_EXSTYLE, style_ex);             // 拡張スタイル変更
    ::SetWindowLong(m_hWnd, GWL_STYLE,   style);                // スタイル変更

    // ウィンドウサイズ設定
    ::AdjustWindowRectEx(&wnd_rect, style, FALSE, style_ex);
    const long   WND_WIDTH  = wnd_rect.right  - wnd_rect.left;  // ウィンドウ幅
    const long   WND_HEIGHT = wnd_rect.bottom - wnd_rect.top;   // ウィンドウ高さ

    // ウィンドウ位置設定(画面の中央に設定)
    const int    POS_X = ※ここは各自考えましょう;        // ウィンドウx座標
    const int    POS_Y = ※ここは各自考えましょう;        // ウィンドウy座標

    // ウィンドウ変更
    ::SetWindowPos(m_hWnd, HWND_TOP, POS_X, POS_Y, WND_WIDTH, WND_HEIGHT,
                   SWP_FRAMECHANGED | SWP_DRAWFRAME | SWP_SHOWWINDOW);

    // オーナーウィンドウ再描画
    HWND   hOwnerWnd = ::GetWindow(m_hWnd, GW_OWNER);
    ::InvalidateRect(hOwnerWnd, NULL, TRUE);
    ::UpdateWindow(hOwnerWnd);

    return true;
}
```

※「オーナーウィンドウ再描画」は、フルスクリーンモードからウィンドウモードに切り替えたときに必要になります。この処理がない場合、一部のウィンドウの再描画処理が正しく行われません。

(４)ウィンドウをモードに合わせて生成するようにします。

CGameWndFrame::Create関数にウィンドウモードで生成するのかを示す引数を追加します。この引数がtrueならウィンドウモード用のウィンドウを、falseならフルスクリーンモード用のウィンドウを生成します。
CGameWndFrame::Create関数のプロトタイプを以下のように変更しましょう。

```
bool Create(const HINSTANCE hInstance, const WNDPROC inWndProc, const bool inWindowed);
```

(５)CGameWndFrame::Create関数をAdjust関数を用いてモードに合わせたウィンドウを生成するように変更します。足りない部分を補い、プログラムを完成させましょう。

```
bool ????????????::Create(const HINSTANCE hInstance, const WNDPROC inWndProc, const bool inWindowed)
{
    Destroy();

    LPCTSTR   WindowClassName = "OOGAMEPROG_WINDOWCLASS";   // ウィンドウクラス名

    WNDCLASSEX   wcx;
    // ウィンドウクラスが登録済みか調べる
    if(::GetClassInfoEx(hInstance, WindowClassName, &wcx) == 0) {
        // ウィンドウクラス登録
        ::ZeroMemory(&wcx, sizeof(wcx));
        wcx.??????        = sizeof(wcx);
        wcx.style         = 0;
        wcx.???????????   = inWndProc;
        wcx.cbClsExtra    = 0;
        wcx.cbWndExtra    = 0;
```

```
        wcx.?????????     = hInstance;
        wcx.hIcon         = ::LoadIcon(NULL, IDI_APPLICATION);
        wcx.hCursor       = ::LoadCursor(NULL, IDC_ARROW);
        wcx.hbrBackground = (HBRUSH)::GetStockObject(BLACK_BRUSH);
        wcx.lpszClassName = WindowClassName;
        wcx.lpszMenuName  = NULL;
        wcx.hIconSm       = NULL;

        if(::?????????????(&wcx) == 0) {
            ::OutputDebugString("*** Error - ウィンドウ登録失敗(CGameWndFrame_Create)\n");
            return false;
        }
    }

    // ウィンドウ生成
    m_hWnd = ::?????????????(0, WindowClassName, CGameApp::APP_NAME, 0,
                             0, 0, 0, 0, NULL, NULL, hInstance, NULL);
    if(m_hWnd == NULL) {
        ::OutputDebugString("*** Error - ウィンドウ生成失敗(CGameWndFrame_Create)\n");
        return false;
    }

    Adjust(inWindowed);                 // ウィンドウ調整
    ::??????????(m_hWnd, SW_SHOW);      // ウィンドウ表示
    ::SetFocus(m_hWnd);                 // フォーカス設定

    // IME消去
    HWND   hIMEWnd = ::ImmGetDefaultIMEWnd(m_hWnd);
    ::SendMessage(hIMEWnd, WM_IME_CONTROL, IMC_CLOSESTATUSWINDOW, 0);

    return true;
}
```

(6)アプリケーションの初期化時に、ウィンドウモードかどうかを指定するようにします。CGameApp::Initialize関数のプロトタイプを以下のように変更しましょう。

```
bool Initialize(const HINSTANCE hInstance, const bool inWindowed);
```

(7)ウィンドウを生成する部分を以下のように変更しましょう。

```
// ウィンドウフレーム生成
if(m_Frame.Create(hInstance, WndProc, inWindowed) == false)
    return false;
```

(8)CGameApp::Initialize関数に渡されたinWindowedを保存するようにします。以下のプログラムをGameApp::Initialize関数の適切な場所に追加しましょう。

```
m_Windowed = inWindowed;   // フラグ保存
```

(9)アプリケーションの初期化部分を以下のように変更し、フルスクリーンモード用のウィンドウになるかを確認しましょう。

```
// アプリケーション初期化
if(App.Initialize(hInstance, false) == false)
    return -1;
```

(10)アプリケーションの初期化部分を以下のように変更し、ウィンドウモード用のウィンドウになるかを確認しましょう。

```
// アプリケーション初期化
if(App.Initialize(hInstance, true) == false)
    return -1;
```

(11)Alt＋Enterキーを押したときにモードが切り替わるようにします。

Alt＋Enterキーが押されたら、モード変更に必要な処理を行います。現時点では、ウィンドウを変更する必要があります。しかし、CGameWndFrameクラスにそのような機能はないので、作成します。
以下のChangeMode関数をCGameWndFrameクラスに追加しましょう。

```
/*****************************************************************************************************/
/*                                          モード変更                                               */
/*****************************************************************************************************/
bool CGameWndFrame::ChangeMode(const bool inWindowed)
{
    return Adjust(inWindowed);      // ウィンドウ調整
}
```

次に、Alt＋Enterキーが押されたときのアプリケーションの動作をまとめて関数にしておきます。
以下のChangeMode関数の足りない部分を補い、適切な場所に追加しましょう。なお、この関数は、外部から呼び出す必要が全くないので、プロトタイプはprivate指定にしましょう。

```
/*****************************************************************************************************/
/*                                          モード変更                                               */
/*****************************************************************************************************/
bool ????????::ChangeMode()
{
    m_Windowed ^= true;                 // モードフラグ反転

    m_Frame.??????????(m_Windowed);     // ウィンドウのモード変更

    return true;
}
```

(12)Alt＋Enterキーが押されたときに発生するメッセージに応答するようにします。以下のプログラムを適切な場所に追加しましょう。

```
      case VK_RETURN:
        ChangeMode();
        return 0;
```