ゲームプログラミング

▶　第４回　いろいろなウィンドウの作成　◀

アプリケーションの目的にあわせたいろいろなウィンドウを生成してみましょう。

ウィンドウ背景色

ウィンドウ背景色は、ウィンドウクラスの定義で使用するWNDCLASSEX構造体のhbrBackgroundメンバで指定します。ウィンドウ背景色は、ブラシを使って塗りつぶされるので、任意のブラシを生成すれば、その属性をウィンドウ背景色にすることができます。

ブラシは、黒・白・灰色・透明といった基本的な色なら定義済みのものがあります。これらのブラシは、GetStockObject関数を呼び出すだけで簡単に生成し、そのハンドルを取得することができます。

任意の色のブラシは、CreateBrushIndirect関数で生成することができます。この関数には、生成するブラシの属性を定義したLOGBRUSH構造体のアドレスを渡します。

LOGBRUSH構造体には、３つのメンバがあります。lbStyleメンバは、ブラシの形状を定義します。BS_SOLID(ソリッドブラシ)、BS_HATCHED(ハッチブラシ)などが指定できます。lbColorメンバは色を定義します。色はRGBマクロで指定します。lbHatchメンバはハッチスタイルを定義します。このメンバはlbStyleメンバがBS_HATCHEDのときに有効になります。HS_CROSS(クロスハッチ)、HS_BDIAGONAL(45度上向きのハッチ)などが指定できます。

生成したブラシは、通常ならDeleteObject関数で解放する必要がありますが、ウィンドウ背景色に使用したブラシは、ウィンドウを破棄するときにWindowsが自動的に解放するので、プログラマが解放する必要はありません。

```
WNDCLASSEX    wcx;

// 定義済みのブラシを使用
wcx.hbrBackground = (HBRUSH)GetStockObject(BLACK_BRUSH);   // 定義済み・黒
wcx.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);   // 定義済み・白


// 任意の色でブラシを生成
LOGBRUSH    lbWndBG;
lbWndBG.lbStyle = BS_SOLID;          // ソリッドブラシ(塗りつぶし)
lbWndBG.lbColor = RGB(255, 0, 0);    // 赤
lbWndBG.lbHatch = 0;

wcx.hbrBackground = CreateBrushIndirect(&lbWndBG);    // オリジナルブラシ


// 任意の色でハッチブラシを生成
LOGBRUSH    lbWndBG;
lbWndBG.lbStyle = BS_HATCHED;        // ハッチブラシ
lbWndBG.lbColor = RGB(0, 255, 0);    // 緑
lbWndBG.lbHatch = HS_CROSS;          // クロスハッチ

wcx.hbrBackground = CreateBrushIndirect(&lbWndBG);    // オリジナルブラシ
```

ウィンドウスタイル

ウィンドウの形状は、CreateWindowEx関数の４つ目の引数で指定します。この引数は、フラグを組み合わせることによりウィンドウの形状が指定できます。フラグは"|"演算子で組み合わせます。フラグによっては組み合わせることができないものがあります。

WS_OVERLAPPEDWINDOW…オーバーラップドウィンドウ。ほとんどのフラグが組み合わされています。
WS_POPUP…ポップアップウィンドウ。クライアント領域のみ。
WS_POPUPWINDOW | WS_CAPTION | WS_MINIMIZEBOX…サイズ変更不可のウィンドウ(境界線＋クライアント領域＋システムメニュー＋タイトルバー＋最小化ボタン)

※フラグによっては、ウィンドウのサイズと座標にCW_USEDEFAULTが使用できなくなります。

拡張ウィンドウスタイル

CreateWindowEx関数の１つ目の引数は拡張ウィンドウスタイルです。WS_EX_TOPMOSTを指定すると常に最前面に表示されるウィンドウになります。このようなウィンドウは、他のアプリケーションのウィンドウに隠されることがなくなります。また、WS_EX_CLIENTEDGEを指定すると縁が沈んだような境界線になります。

クライアント領域のサイズを指定

ウィンドウのサイズは、CreateWindowEx関数の７つ目と８つ目の引数で指定します。ゲームでは、６４０×４８０ピクセルというサイズがよく使用されます。６４０×４８０ピクセルの描画領域をもったウィンドウを生成するには、以下のように記述したくなります。

```
HWND   hWnd = CreateWindowEx(0, MAINWND_CLSNAME, "Windowsプログラミング",
                             WS_POPUPWINDOW | WS_CAPTION | WS_MINIMIZEBOX,
                             CW_USEDEFAULT, CW_USEDEFAULT,
                             640, 480,
                             HWND_DESKTOP, NULL,
                             hInstance,    NULL);
```

ところが、ここでいうウィンドウのサイズとは、ウィンドウ全体のサイズなので、境界線やタイトルバーなどを含めて６４０×４８０ピクセルになります。グラフィックを描画するためのクライアント領域はこれより小さくなってしまいます。

クライアント領域のサイズを指定したい場合は、AdjustWindowRectEx関数で正しいウィンドウ全体のサイズを求めます。RECT構造体にクライアント領域のサイズを定義しておき、AdjustWindowRectEx関数の１つ目の引数にそのアドレスを渡します。２つ目の引数は生成するウィンドウのスタイルです。３つ目の引数はメニューの有無で、TRUE(メニューあり)、FALSE(メニューなし)です。４つ目の引数は拡張ウィンドウスタイルです。これらの引数をもとに、ウィンドウ全体のサイズが計算されます。関数が成功すると、１つ目の引数の内容が書き換えられます。ウィンドウ幅はrightメンバからleftメンバを引くと求まり、ウィンドウの高さはbottomメンバからtopメンバを引くことで求めることができます。

```
// AdjustWindowRectEx関数で必要なウィンドウサイズを求める
RECT    rcWndSize  = {0, 0, 640, 480};                          // クライアント領域のサイズ
DWORD   dwWndStyle = WS_POPUPWINDOW | WS_CAPTION | WS_MINIMIZEBOX; // ウィンドウスタイル
if(0 == AdjustWindowRectEx(&rcWndSize, dwWndStyle, FALSE, 0))
    return 0;

const long   lWndWidth  = rcWndSize.right  - rcWndSize.left;    // ウィンドウの幅を求める
const long   lWndHeight = rcWndSize.bottom - rcWndSize.top;     // ウィンドウの高さを求める

// メインウィンドウ生成
HWND   hWnd = CreateWindowEx(0, MAINWND_CLSNAME, "Windowsプログラミング",
                             dwWndStyle,
                             CW_USEDEFAULT, CW_USEDEFAULT,
                             lWndWidth, lWndHeight,
                             HWND_DESKTOP, NULL,
                             hInstance,    NULL);
```

○ 課題 ○

(１)フルスクリーンで動作するゲーム向けのウィンドウを作成しましょう。

このようなアプリケーションのウィンドウは、常に最前面に表示され、画面全体を覆うようにします。ウィンドウ背景色は、黒または透明にすると綺麗に見えます。ウィンドウ表示位置は(０，０)、クライアント領域のみを持つウィンドウにします。ウィンドウのサイズは、ディスプレイ(スクリーン)と同じサイズにして画面全体を覆います。

ディスプレイのサイズはそのときどきによって異なりますが、GetSystemMetrics関数で取得することができます。この関数は、引数に取得したい表示要素を指定します。ただし、SM_CXFULLSCREENとSM_CYFULLSCREENはウィンドウを最大化したときのサイズを取得するものなので、正確なサイズを取得できるとは限りません。

(２)ウィンドウモードで動作するゲーム向けのウィンドウを作成しましょう。

このようなアプリケーションのウィンドウは、基本的には通常のアプリケーションのウィンドウと大差ありません。ウィンドウ背景色は任意の色、表示位置は画面の中央にくるようにします。ウィンドウスタイルは境界線(縁が沈んだもの)、クライアント領域、システムメニュー、タイトルバー、最小化ボタンを持ち、ウィンドウサイズの変更ができないものにします。

ヒント