```
//
// list203.cpp : コンソール アプリケーションのエントリ ポイントを定義します。
//
#include <stdio.h>
#include <conio.h>          // getch()の宣言
#include <windows.h>
#include <process.h>        // スレッド用

// このコード モジュールに含まれる関数の宣言を転送します :
DWORD WINAPI        thProc(LPVOID);
LRESULT CALLBACK    WndProc(HWND, UINT, WPARAM, LPARAM);

HINSTANCE hInstance;        // スレッド側でも利用するのでグローバル変数とする

void main()
{
    // WinMain()と異なりmain()はシステムからインスタンス・ハンドルを得られない
    // 代わりに指定されたウィンドウに関する情報を取得するGetWindowLong()を使う
    hInstance = (HINSTANCE)GetWindowLong(HWND_DESKTOP, GWL_HINSTANCE);
    WNDCLASSEX wcex;        // 新しくつくるウィンドウクラス用構造体
    memset(&wcex, 0, sizeof(WNDCLASSEX));
    wcex.cbSize = sizeof(WNDCLASSEX);   // WNDCLASSEXの大きさ
    wcex.lpfnWndProc    = (WNDPROC)WndProc;// このクラスの持つウィンドウプロシー
ジャ
    wcex.hInstance      = hInstance;
    wcex.hCursor        = LoadCursor(NULL, IDC_ARROW);
    wcex.hbrBackground  = (HBRUSH)GetStockObject(WHITE_BRUSH);
    wcex.lpszClassName  = "wc00"; // szWindowClass;

    if (!RegisterClassEx(&wcex)) return;    // ウィンドウクラスの登録

    // メッセージループのスレッドを起動
    DWORD id;
    CreateThread(NULL,      // pointer to thread security attributes
        0,                  // initial thread stack size, 0はデフォルト
        thProc,             // pointer to thread function
        NULL,               // argument for new thred
        0,                  // creation flags
        &id);               // pointer to returned thread identifier

    printf("¥nコンソールからスレッドを使ってウィンドウを作成¥n");
    printf("終了するには何かキーを押してください。¥n¥n");
    getch();
}
```

```
/**
 *  メッセージ・ループのためのスレッド、ここでウィンドウを作る
 */
DWORD WINAPI thProc(LPVOID)
{
    MSG msg;

    CreateWindow("wc00",                    // ウィンドウクラスの名前
        "コンソール&マルチスレッド版 hello", // キャプションの内容
        WS_OVERLAPPEDWINDOW | WS_VISIBLE,   // ウィンドウの属性、クラスの名前
        CW_USEDEFAULT, CW_USEDEFAULT,       // 位置は指定しない
        400, 300,                           // ウィンドウの大きさ
        HWND_DESKTOP,                       // 親はデスクトップ
        NULL, hInstance, NULL); // メニューハンドルなし、インスタンス、

    while (GetMessage(&msg, NULL, 0, 0))    // メッセージループ
    {
        DispatchMessage(&msg);
    }

    return 0;
}

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    switch (message)
    {
    case WM_PAINT:
        {
        PAINTSTRUCT ps;
        char *p = "hello windows";
        BeginPaint(hWnd, &ps);
        TextOut(ps.hdc, 10, 10, p, (int)strlen(p));
        EndPaint(hWnd, &ps);
        break;
        }
    case WM_DESTROY:
        PostQuitMessage(0);
        break;
    default:
        return DefWindowProc(hWnd, message, wParam, lParam);
    }
    return 0;
}
```