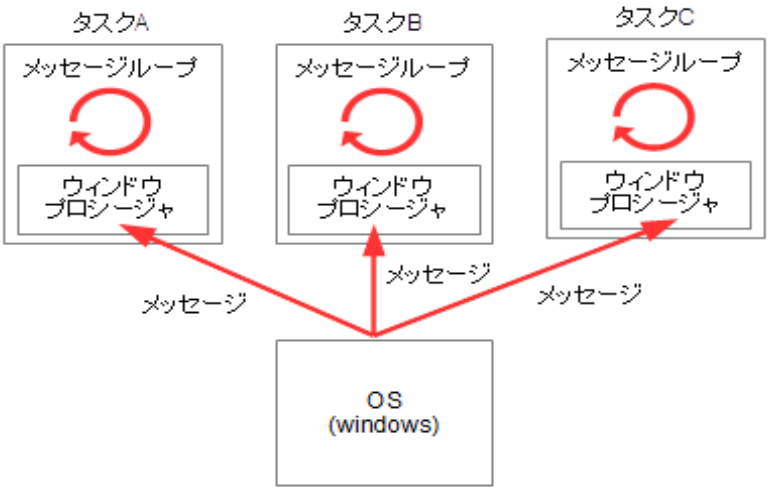
【１】DXライブラリを使用せず、Windows APIを使用してウィンドウプログラムを作成

①ウィンドウアプリとOS(Windows)の関係性
既に分かっている事とは思いますが、Windowsに限らず、現在のほとんどのコンピュータのOSは、マルチタスクOSと呼ばれるものに分類されます。
マルチタスクとは、複数の仕事を同時にこなす、という意味ですね。例えばウィンドウズでは、ウェブブラウザと、ワープロソフトと、メールソフトなどといったソフトを同時並行に使うことができます。これが、マルチタスクという事です。

Windows は、これら複数のアプリケーションを管理しているわけですが、基本的にこれらの各アプリケーションはそれぞれ別々に動作しています。それらに対しOSは、メッセージと呼ばれる指令を与えます。例えば、メールソフトが特に操作をしなくても新着メールが来ると画面に表示され効果音が鳴ったり、ウェブブラウザで動画投稿サイトで動画が出力されるのは、すべてWindowsがこのメッセージを送っているからです。

このようなスタイルのアプリケーションの事を、**メッセージ駆動型**と言います。
メッセージ駆動型のアプリの特徴は、OSから送られるメッセージによって処理内容が変わるという事です。



ウインドウは常にメッセージが発生しないかどうかを、**メッセージループ**で監視しています。
これに対し、実際に何かメッセージが発せられた時に実行される関数をウィンドウプロシージャと呼びます。

キー操作やマウス操作もまたメッセージの一つです。例えば、マウスをウインドウ上でクリックすると、左クリックされましたよ！というメッセージがウインドウプロシージャへ送られます。
メッセージを受け取った側に、あらかじめ左クリックした場合の処理が記述してあれば、それに応じた処理を行います。なければスルーします。

②実際に簡単なウィンドウを表示するプログラムを作ってみましょう。

main.cpp

```
//-------------------------
// main.cpp
//-------------------------
#include <Windows.h>
#include <WinUser.h>
#include <stdlib.h>
#include <string.h>
#include <tchar.h>
#include <string>

static constexpr TCHAR tszAplName[] = "step01";

static constexpr int WINDOW_WID = 800;
static constexpr int WINDOW_HIG = 600;

// ウィンドウプロシージャのコールバック関数のプロトタイプ宣言
LRESULT CALLBACK WndProc(HWND hWnd, UINT msg, WPARAM wParam, LPARAM lParam);

int WINAPI WinMain(HINSTANCE hInst, HINSTANCE hPrevInst, LPSTR lpCmdLine, int nCmdShow)
{
  /**
   *-----------------------------------------------
   * Windowの作成と表示
   *-----------------------------------------------
   */
  WNDCLASSEX wndcex;

  ZeroMemory(&wndcex, sizeof(wndcex));
  wndcex.cbSize = sizeof(WNDCLASSEX);
  wndcex.style = CS_BYTEALIGNCLIENT | CS_VREDRAW | CS_HREDRAW;
  wndcex.lpfnWndProc = WndProc;
  wndcex.cbClsExtra = 0;
  wndcex.cbWndExtra = 0;
  wndcex.hInstance = hInst;
  wndcex.hIcon = LoadIcon(hInst, MAKEINTRESOURCE(IDI_APPLICATION));
  wndcex.hCursor = NULL;
  wndcex.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
  wndcex.lpszMenuName = NULL;
  wndcex.lpszClassName = tszAplName;
  wndcex.hIconSm = LoadIcon(wndcex.hInstance, MAKEINTRESOURCE(IDI_APPLICATION));

  if (!RegisterClassEx(&wndcex)) {
    return false;
  }
```

```
   HWND hwnd = CreateWindowEx(0, tszAplName, tszAplName,
      WS_OVERLAPPED | WS_SYSMENU | WS_CAPTION | WS_THICKFRAME | WS_MAXIMIZEBOX,
      CW_USEDEFAULT, CW_USEDEFAULT, WINDOW_WID, WINDOW_HIG,
      NULL, NULL, hInst, NULL);

   if (hwnd == NULL)return false;

   ShowWindow(hwnd, nCmdShow);
   UpdateWindow(hwnd);


   MSG msg;
   while (true) {
      if (PeekMessage(&msg, NULL, 0, 0, PM_REMOVE)) {
         //if (!GetMessage(&msg, NULL, 0, 0))break;
         if (msg.message == WM_QUIT)break;

         TranslateMessage(&msg);
         DispatchMessage(&msg);
      }
   }

   return (int)msg.wParam;
}

/**
 *---------------------------------------------
 * メインウィンドウPROC
 *---------------------------------------------
 */
LRESULT CALLBACK WndProc(HWND hWnd, UINT msg, WPARAM wParam, LPARAM lParam)
{
   PAINTSTRUCT ps;
   HDC hdc;
   std::string str = "Hello World";

   switch (msg) {
   case WM_PAINT:
      // 描画処理の開始
      hdc = BeginPaint(hWnd, &ps);
      TextOut(hdc, 5, 5, str.c_str(), (int)strlen(str.c_str()));
      EndPaint(hWnd, &ps);
      break;
   case WM_DESTROY:
      PostQuitMessage(0);
      break;
   default:
      return DefWindowProc(hWnd, msg, wParam, lParam);
   }

   return 0;
}
```

実行するとこの様なウィンドウが表示されます。