

МИНИСТЕРСТВО ОБРАЗОВАНИЯ ОРЕНБУРГСКОЙ ОБЛАСТИ
Государственное автономное профессиональное образовательное
учреждение
**«ОРЕНБУРГСКИЙ КОЛЛЕДЖ ЭКОНОМИКИ И ИНФОРМАТИКИ»
(ГАПОУ ОКЭИ)**

Практическая работа

ОКЭИ 09.02.07 7026 07 У

Тема: *«Разработка оконного приложения. Работа с сообщениями»*

Выполнил: *Гадияев Ислам Ильгамович*

Оренбург 2026

Содержание

1 Цель работы 3

2 Скриншот структуры готового проекта..... 3

3 Листинг программы 3

4 Результаты работы программы..... 5

					ОКЭИ 09.02.07 7026 07 У									
Изм.	Лист	№ докум.	Подп.	Дата										
Разраб.	Гадилыев И. И.				Практическая работа						Лит.	Лист	Листов	
											У		2	
											Отделение программирования			

1 Цель работы

Задание: в ранее разработанное приложение(пз_14) добавить реакцию на сообщения: WM_ACTIVATE, WM_MOVE, WM_RBUTTONDOWNBLCLK (для визуализации ответа на сообщение можно поменять цвет заливки окна)

2 Скриншот структуры готового проекта

На рисунке 1 представлен скриншот структуры готового проекта.

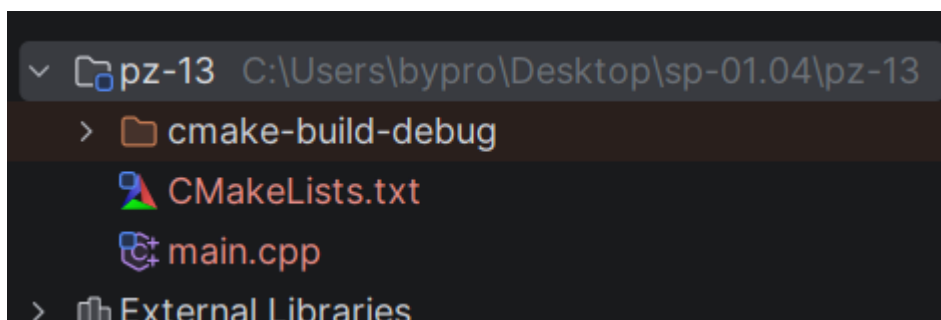


Рисунок 1 – Структура проекта

3 Листинг программы

Листинг файла main.cpp представлен ниже.

```
#define STRICT
#define WIN32_LEAN_AND_MEAN

#include <windows.h>

LRESULT CALLBACK WindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    const char* CLASS_NAME = "MyWindowsClass";

    WNDCLASSEX wc = {};
    wc.cbSize = sizeof(WNDCLASSEX);
    wc.style = CS_HREDRAW | CS_VREDRAW | CS_DBLCLKS; // добавили CS_DBLCLKS
    для двойных кликов (в т.ч. правой кнопкой)
    wc.lpfnWndProc = WindowProc;
    wc.hInstance = hInstance;
    wc.hIcon = LoadIconA(nullptr, IDI_APPLICATION);
    wc.hCursor = LoadCursorA(nullptr, IDC_ARROW);
    wc.hbrBackground = (HBRUSH)GetStockObject(BLACK_BRUSH);
    wc.lpszClassName = CLASS_NAME;
    wc.hIconSm = LoadIconA(nullptr, IDI_APPLICATION);

    if (!RegisterClassExA(&wc)) {
        MessageBoxA(nullptr, "Register failed", "Error", MB_OK);
        return 0;
    }
}
```

```

    }

    HWND hWnd = CreateWindowExA(
        0,
        CLASS_NAME,
        "Hello1",
        WS_OVERLAPPEDWINDOW,
        CW_USEDEFAULT, CW_USEDEFAULT,
        800, 600,
        nullptr, nullptr, hInstance, nullptr
    );

    if (!hWnd) {
        MessageBoxA(nullptr, "CreateWindow failed", "Error", MB_OK);
        return 0;
    }

    ShowWindow(hWnd, nCmdShow);
    UpdateWindow(hWnd);

    MSG msg = {};
    while (GetMessageA(&msg, nullptr, 0, 0)) {
        TranslateMessage(&msg);
        DispatchMessageA(&msg);
    }

    return (int)msg.wParam;
}

LRESULT CALLBACK WindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    static HBRUSH currentBrush = (HBRUSH)GetStockObject(BLACK_BRUSH);

    switch (uMsg) {
        case WM_ACTIVATE:
        {
            HBRUSH newBrush;
            if (LOWORD(wParam) == WA_INACTIVE) {
                newBrush = CreateSolidBrush(RGB(100, 100, 100));
            } else {
                newBrush = CreateSolidBrush(RGB(0, 255, 0));
            }

            if (currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
                currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
                currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
                DeleteObject(currentBrush);
            }

            currentBrush = newBrush;
            SetClassLongPtrA(hWnd, GCLP_HBRBACKGROUND, (LONG_PTR)newBrush);
            InvalidateRect(hWnd, nullptr, TRUE);
            return 0;
        }

        case WM_MOVE:
        {
            HBRUSH newBrush = CreateSolidBrush(RGB(0, 0, 255));

            if (currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
                currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
                currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
                DeleteObject(currentBrush);
            }

            currentBrush = newBrush;
            SetClassLongPtrA(hWnd, GCLP_HBRBACKGROUND, (LONG_PTR)newBrush);
            InvalidateRect(hWnd, nullptr, TRUE);
            return 0;
        }
    }
}

```

```

    }

    case WM_RBUTTONDOWNLCLK:
    {
        HBRUSH newBrush = CreateSolidBrush(RGB(255, 0, 0));

        if (currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
            currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
            currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
            DeleteObject(currentBrush);
        }

        currentBrush = newBrush;
        SetClassLongPtrA(hWnd, GCLP_HBRBACKGROUND, (LONG_PTR)newBrush);
        InvalidateRect(hWnd, nullptr, TRUE);
        return 0;
    }

    case WM_DESTROY:
    {
        if (currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
            currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
            currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
            DeleteObject(currentBrush);
        }
        PostQuitMessage(0);
        return 0;
    }

    default:
        return DefWindowProcA(hWnd, uMsg, wParam, lParam);
}
}

```

4 Результаты работы программы

На рисунке 2 представлено окно, когда оно в неактив.

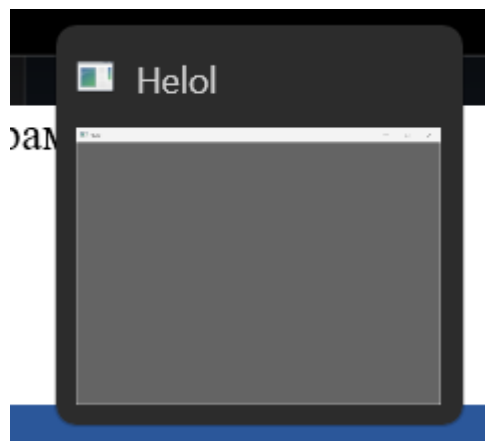


Рисунок 2 – Окно в неактив

На рисунке 3 представлено окно, когда оно открыто.



Рисунок 3 – Открытое окно

На рисунке 4 представлено окно, когда оно перетаскивается.



Рисунок 4 – Перетаскивается окно

