Universitatea Tehnică a Moldovei
Facultatea de Calculatoare, Informatică şi Microelectronică
Filiera Anglofonă „Computer Science”

Aplicații Windows
Raport pentru Laboratorul #1

Windows Applications
Laboratory Work #1 Report

Student:

gr. FAF-111
Mașaev Iulian

Conducător:

lector assistent
Truhin Alexandr

Chișinău 2013

Introduction

In this laboratory work I had to write a C code for generating a window using functions from Win32 API, and write there in the middle my name. Also additionally I should change some behavior of window actions at my wish.

Theoretical aspects of laboratory work

Win32 API is a set of functions, which can be called by program, but performed by the system. In libraries we have prototypes, constants, data structures, that are actually defined in Windows installation directory. We don't have to write code for them, but using them we can define any window we want.

Windows communicate with another windows and with Operating System using messages. A message is a structure, with fields:

```
HWND      hwnd       //window handle
UINT      message    //Unsigned int(32bits) – code of msg
WPARAM    wParam     //A 32-bit parameter of the msg
LPARAM    lParam     //another one parameter of the msg
DWORD     time       //time when msg was placed in queue
POINT     pt         //coordinates of mouse at that time
```

Window handle is a number defined by the operating system to each window, even if this is just a button, and is used by It to identify target window of the message, or from which window it comes. Actions after each message is defined by the programmer, usually in a switch-case construction.

But what is a window? In WinAPI, it is a class, because it has data, which consists of different parameters of the window, and functions, defined by OS and user. For working with windows, we need to declare a WNDCLASS, of type WNDCLASS, with different fields, including pointer to window procedure, style of the window, size, position, instance handle of the program, icons, class name and so one. After that we can register class, calling RegisterClass(&wndclass), and afterwards create how many windows we want, based on this class, of course with different parameters; anyway they all will have different handles, so it will be possible to differentiate them. After that window needs be shown on the display, and updated, for painting client zone.

Now we include message loop. It may look strange at first sight, but since in each message there Is a handle to window, and each window is defined on a class, which contain a pointer to window procedure. In our case it is called WndProc. It has parameters: HWND handle to window, UINT message, and 2 more parameters of the message.

Implementation description

For implementing this tasks, I just used structures and functions that are used in most WinAPI applications, defining/changing and experimenting with them. For processing messages I tried to understand how to get information from parameters, but unfortunately I didn't found any example or explanation, so I just introduced 2 message boxes. In winuser.h there are different constants with name WM_*, where WM stands for window message, with intuitive names, what is left is to call appropriate function, and use info from parameters of the message.

Conclusions

WinAPI could be a easy tool if used by experienced programmer. The work flow is quite simple at the beginning – define class, create one window, and then process every message which comes from the window. The code for this is common for almost all windows application, what we can do – define each parameter as we want. But for me is necessary to learn more about Windows Programming, because I understood a little bit how to limit window size, but get in troubles when I tried to access info from lParam, where should be pointer to a structure RECT, and math behind this is not complicated, just compare cx and cy with constants, and decide – to resize the windows, or to leave it's size unchanged. Also I wish to learn about fetching mouse pressing on buttons.

Bibliography

Charles Petzold, Programming Windows, Win32 API, 5th edition

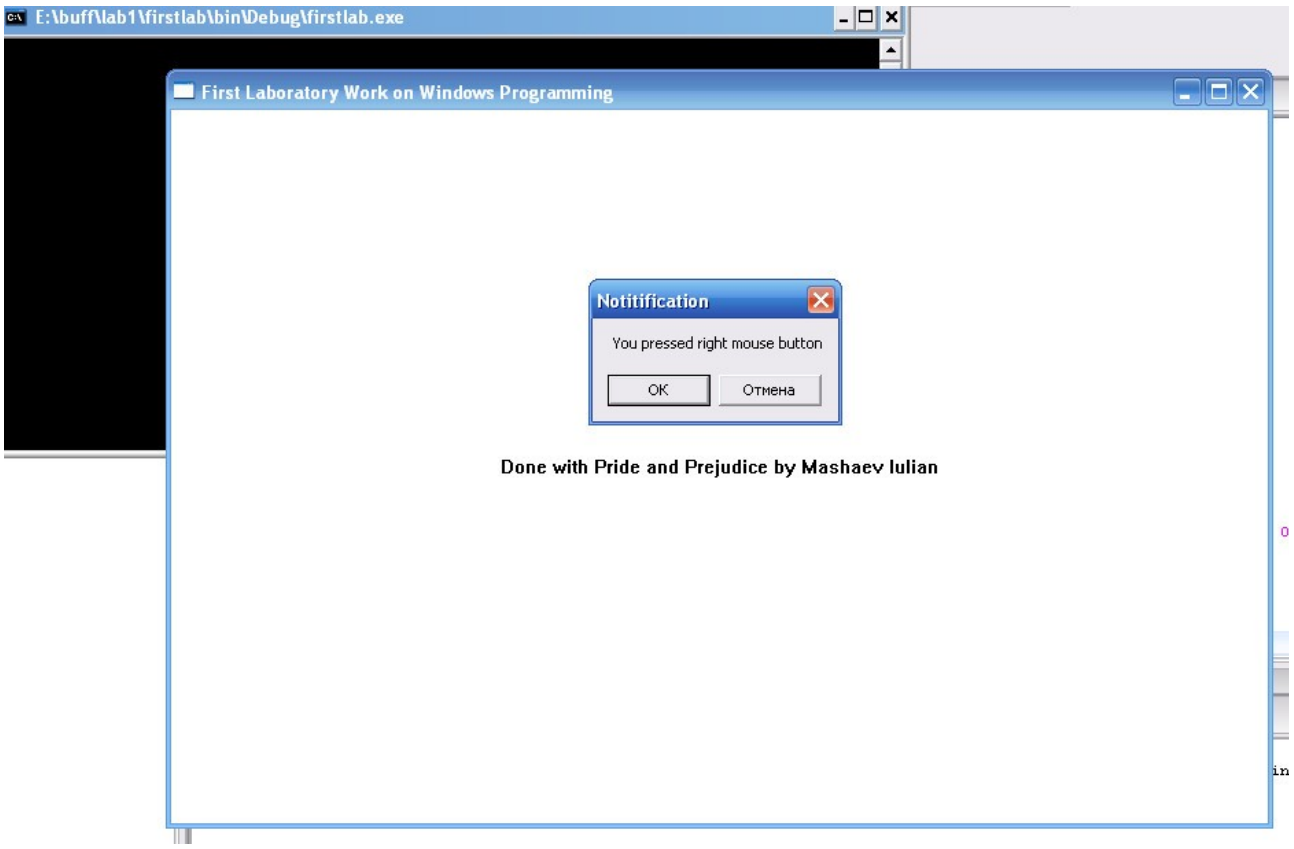
winAPI libraries

Appendices

Appendix 1: C code

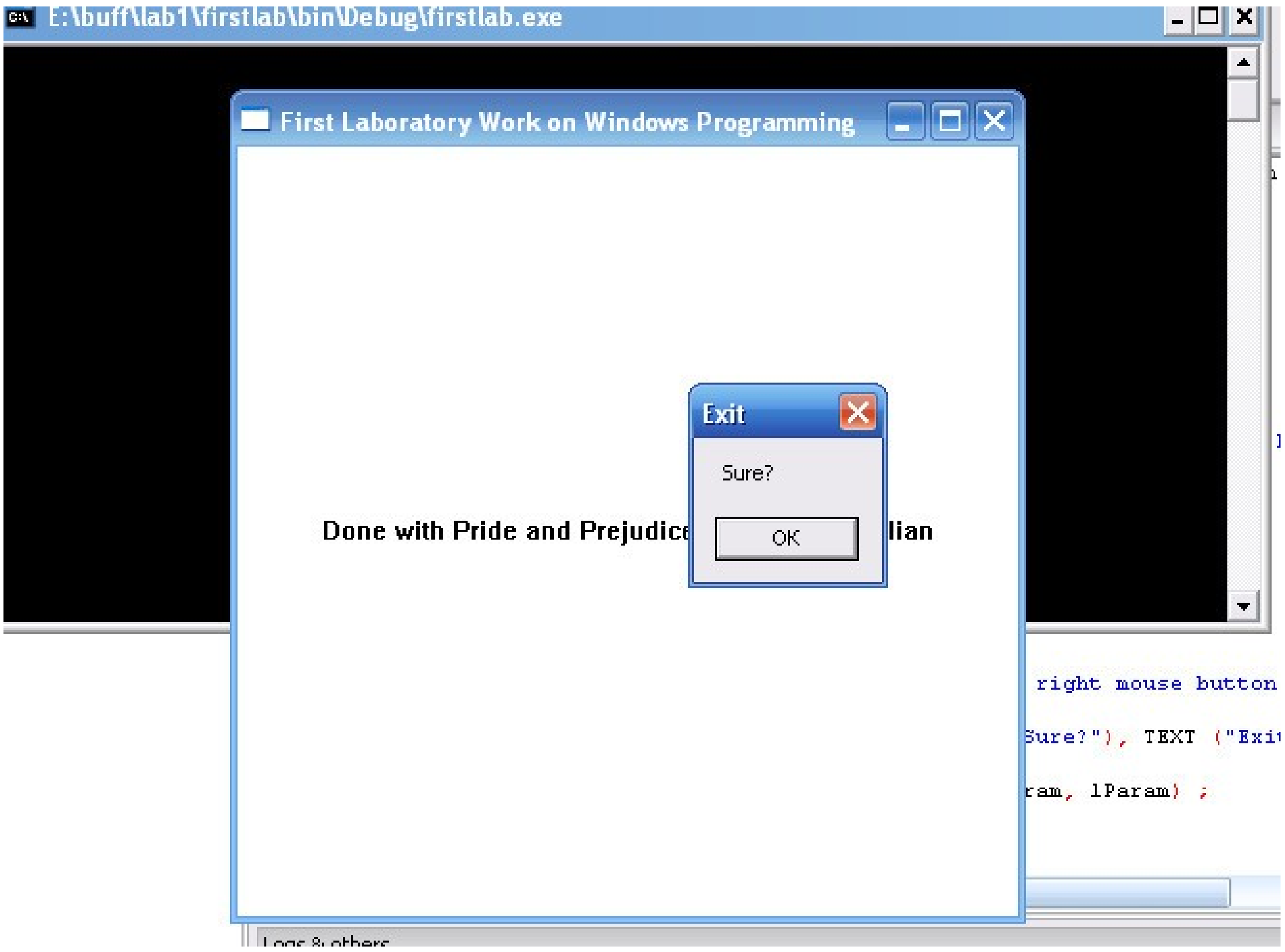
```
#include<windows.h>
LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR szCmdLine,
int iCmdShow)
{
    static TCHAR szAppName[] = TEXT("labwork");
    HWND     hwnd;
    MSG      msg;
    WNDCLASS wndclass;

    wndclass.style = CS_HREDRAW | CS_VREDRAW ;
    wndclass.lpfnWndProc = WndProc;
    wndclass.cbClsExtra = 0;
    wndclass.cbWndExtra = 0;
    wndclass.hInstance = hInstance;
    wndclass.hIcon = LoadIcon(NULL, IDI_APPLICATION);
    wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);
    wndclass.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
    wndclass.lpszMenuName = NULL;
    wndclass.lpszClassName = szAppName;

    RegisterClass (&wndclass);

      hwnd = CreateWindow(szAppName, TEXT ("First Laboratory Work on Windows
Programming"), WS_OVERLAPPEDWINDOW,
                                   CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
CW_USEDEFAULT, NULL, NULL, hInstance, NULL);

    ShowWindow(hwnd, iCmdShow);
    UpdateWindow(hwnd);

    while(GetMessage(&msg, NULL, 0, 0))
    {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    return msg.wParam ;
}

LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    HDC         hdc ;
    PAINTSTRUCT ps ;
    RECT        rect ;
    switch (message)
    {
    case WM_PAINT:
        hdc = BeginPaint (hwnd, &ps) ;
        GetClientRect (hwnd, &rect) ;
        DrawText (hdc, TEXT ("Done with Pride and Prejudice by Mashaev Iulian"),
-1, &rect,
                  DT_SINGLELINE | DT_CENTER | DT_VCENTER ) ;
        EndPaint (hwnd, &ps) ;
        return 0 ;

    case WM_DESTROY:
        PostQuitMessage (0) ;
        return 0 ;
    case WM_RBUTTONDOWN:
             MessageBox (NULL, TEXT ("You pressed right mouse button"), TEXT
("Notitification"), 1) ;
        return 0;
       case WM_CLOSE: MessageBox (NULL, TEXT ("Sure?"), TEXT ("Exit"), 0) ;
PostQuitMessage (0) ;return 0;
    }
    return DefWindowProc (hwnd, message, wParam, lParam) ;
}
```

Appendix 2: Screen shoots


E:\buff\lab1\firstlab\bin\Debug\firstlab.exe
First Laboratory Work on Windows Programming
Notitification
OK
Отмена

You pressed right mouse button

Done with Pride and Prejudice by Mashaev Iulian


E:\buff\lab1\firstlab\bin\Debug\firstlab.exe
First Laboratory Work on Windows Programming
Exit
OK
Logs & others

Sure?

Done with Pride and Prejudice ... lian

```
right mouse button
Sure?"), TEXT ("Exi
ram, lParam) ;
```