Windows API GUIs 1: Example code and tutorial (for the above lecture)

Building Graphical User Interfaces (GUIs)

Using the Windows Application Programming Interface (API)

All of these examples assume that Microsoft Visual C++ 2017 is the compiler being used. Based on a tutorial written by Dr. Rick Coleman.

Exercise 1: A very simple first GUI program

There is a tradition among C programmers that your first program prints "Hello, world!" on the screen. Here you will follow that tradition.

1. Create a project as shown in the lecture video
2. Add a .cpp source file to the project.
3. Add paste in the following code:

```
#include <windows.h>

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
        LPSTR lpCmdLine, int nCmdShow)
{
        MessageBox(NULL, "Hello, world!", "Note", MB_OK | MB_ICONEXCLAMATION);
        return 0;
}
```

4. Compile and run the application. You should see a small message box like the one in the lecture video.

What's it all mean?

1. **#include <windows.h>** – includes the basic Windows library
2. **WINAPI** – specifies the calling convention. Don’t worry about it. Just accept that it is needed for a Windows app.
3. **WinMain()** – Windows equivalent of main()
4. **HINSTANCE hInstance** – Handle to the program's executable module (the .exe file in memory). You will see how to use this later. For now just ignore it.
5. **HINSTANCE hPrevInstance** – Always NULL for Win32 programs. Used to be the handle to the previously run instance of your program (if any) in Win16. This no longer applies. In Win32 you ignore this parameter.
6. **LPSTR lpCmdLine** – The command line arguments, if any, as a single string. NOT including the program name.
7. **int nCmdShow** – An integer value which may be passed to ShowWindow(). You will get to this later.
8. **MessageBox()** – Function to call to display a simple message box. The args are:
    - a. **NULL** – Means there is no parent window owning this dialog.

- b. **"Hello, world!"** – Message to display. If you didn't set the Character Set to "Not Set" when creating the project as explained above then you have to put an **L** in front of the opening quote. The **L** means that this string should be interpreted as a wide char (wchar_t or Unicode character) string instead of a plain char string. MessageBox expects this.
- c. **"Note"** – Title of the window appearing in the title bar.
- d. **MB_OK | MB_ICONEXCLAMATION** – A flag which does a bitwise **OR** of two flag values to tell MessageBox to display an OK button and use the exclamation point icon.

Now try modifying this "Hello, World" code:

1. Change the text of the message and note that the message box always resizes to accomodate the text.
2. Try inserting the '\n' character to create multi-line messages.
3. Try different combinations for the fourth argument to the MessageBox function call
    - **MB_ABORTRETRYIGNORE -** The message box contains three push buttons: Abort, Retry, and Ignore.
    - **MB_CANCELTRYCONTINUE -** Microsoft Windows 2000/XP: The message box contains three push buttons: Cancel, Try Again, Continue. Use this message box type instead of MB_ABORTRETRYIGNORE.
    - **MB_HELP -** Windows 95/98/Me, Windows NT 4.0 and later: Adds a Help button to the message box. When the user clicks the Help button or presses F1, the system sends a WM_HELP message to the owner.
    - **MB_OK -** The message box contains one push button: OK. This is the default.
    - **MB_OKCANCEL -** The message box contains two push buttons: OK and Cancel.
    - **MB_RETRYCANCEL -** The message box contains two push buttons: Retry and Cancel.
    - **MB_YESNO -** The message box contains two push buttons: Yes and No.
    - **MB_YESNOCANCEL -** The message box contains three push buttons: Yes, No, and Cancel. To display an icon in the message box, specify one of the following values.
    - **MB_ICONEXCLAMATION -** An exclamation-point icon appears in the message box.
    - **MB_ICONWARNING -** An exclamation-point icon appears in the message box.
    - **MB_ICONINFORMATION -** An icon consisting of a lowercase letter i in a circle appears in the message box.
    - **MB_ICONASTERISK -** An icon consisting of a lowercase letter i in a circle appears in the message box.
    - **MB_ICONQUESTION -** A question-mark icon appears in the message box. The question-mark message icon is no longer recommended because it does not clearly represent a specific type of message and because the phrasing of a message as a question could apply to any message type. In addition, users can confuse the message symbol question mark with Help information. Therefore, do not use this question mark message symbol in your message boxes. The system continues to support its inclusion only for backward compatibility.
    - **MB_ICONSTOP -** A red circle icon with an X on it appears in the message box and a warning tone is sounded.
    - **MB_ICONERROR -** A red circle icon with an X on it appears in the message box and a warning tone is sounded.
    - **MB_ICONHAND -** A red circle icon with an X on it appears in the message box and a warning tone is sounded.

    To indicate the default button, specify one of the following values.

    - **MB_DEFBUTTON1 -** The first button is the default button. MB_DEFBUTTON1 is the default unless MB_DEFBUTTON2, MB_DEFBUTTON3, or MB_DEFBUTTON4 is specified.
    - **MB_DEFBUTTON2 -** The second button is the default button.
    - **MB_DEFBUTTON3 -** The third button is the default button.
    - **MB_DEFBUTTON4 -** The fourth button is the default button.

For more information about the various flags and use of MessageBox check Microsoft's **description of the MessageBox (http://msdn.microsoft.com/en-us/library/ms645505(VS.85).aspx)** function.