计算机图形学 实验课
Computer Graphics Practicum

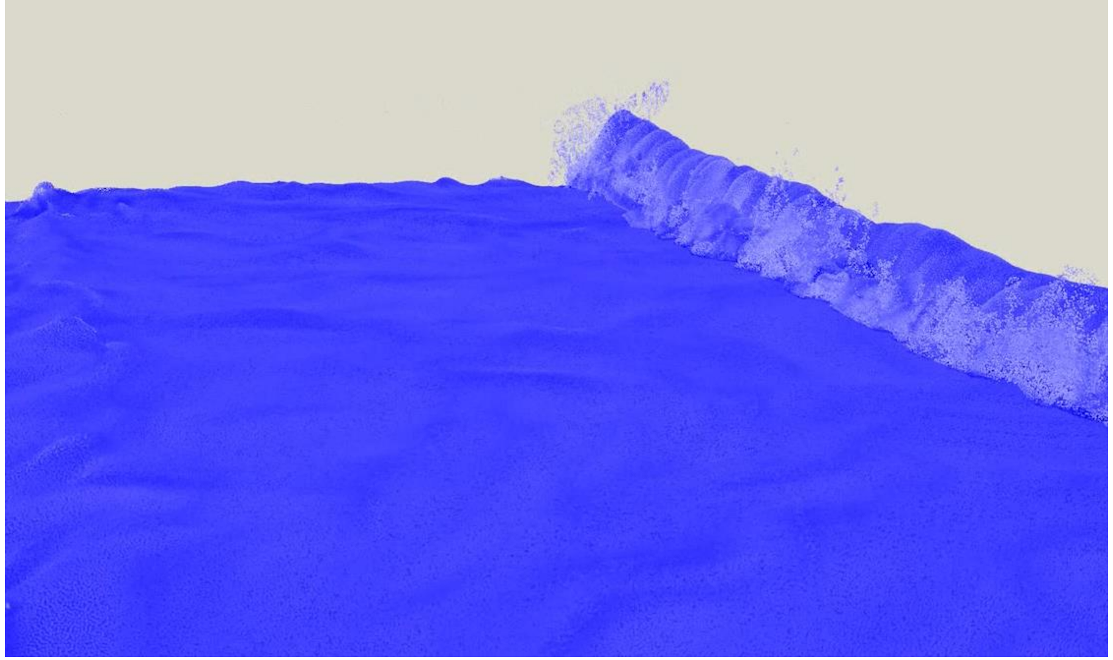
华东师范大学计算机科学与技术学院

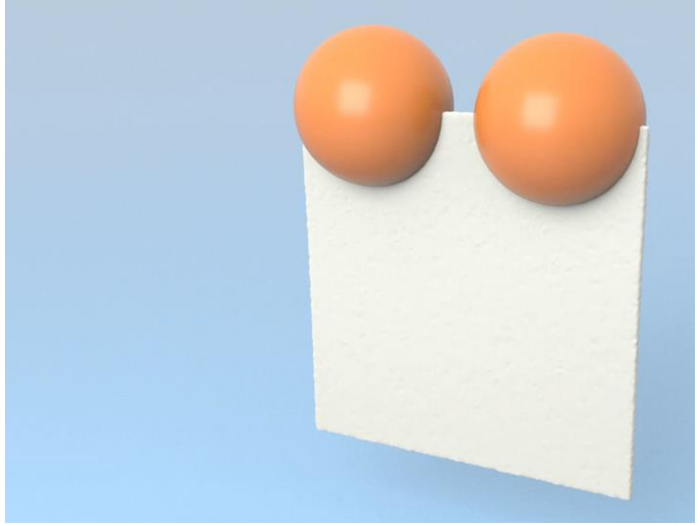
李晨 副研究员

cli@cs.ecnu.edu.cn

About me

- Personal homepage
  - faculty.ecnu.edu.cn/_s16/lc2/main.psp
- Research interests
  - Computer Graphics
  - Geometric and Physics-based Modeling
  - Computer Animation and Simulation
  - Deep Learning and Differentiable Physics
- Office
  - Room B804, 09:00-18:00
  - cli@cs.ecnu.edu.cn





Course Information

- 08:00-09:35, B517
- Tencent meeting: 923-8244-4395, password: 2022
- TA:刘龙
- E-mail: 51255901037@stu.ecnu.edu.cn

In this course

- **You will**
  - explore fundamental ideas in computer graphics
  - implement simple yet key algorithms
  - learn the basics of GDI/OpenGL
- **You will not**
  - write very big programs
  - learn much for the modern development platform, e.g., Unity, Unreal
  - learn the 3D modeling, rendering, and animation software, e.g., 3ds Max, blender

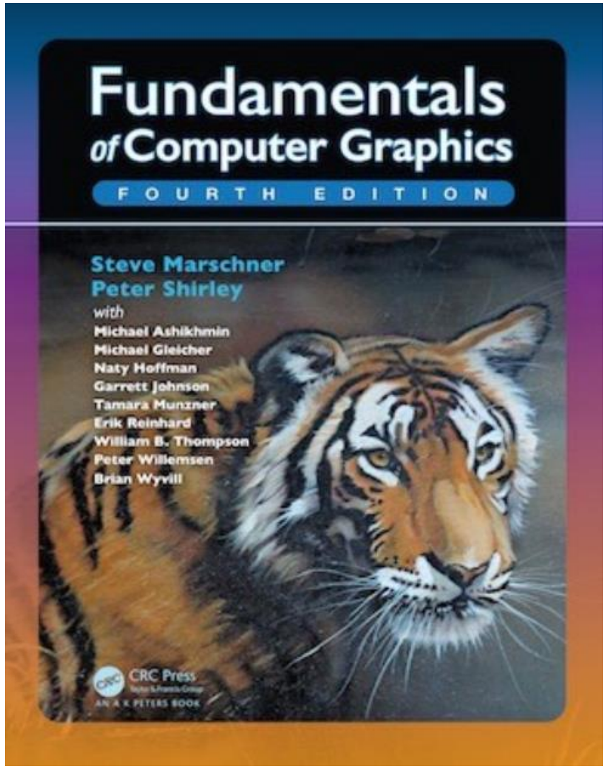
Prerequisites

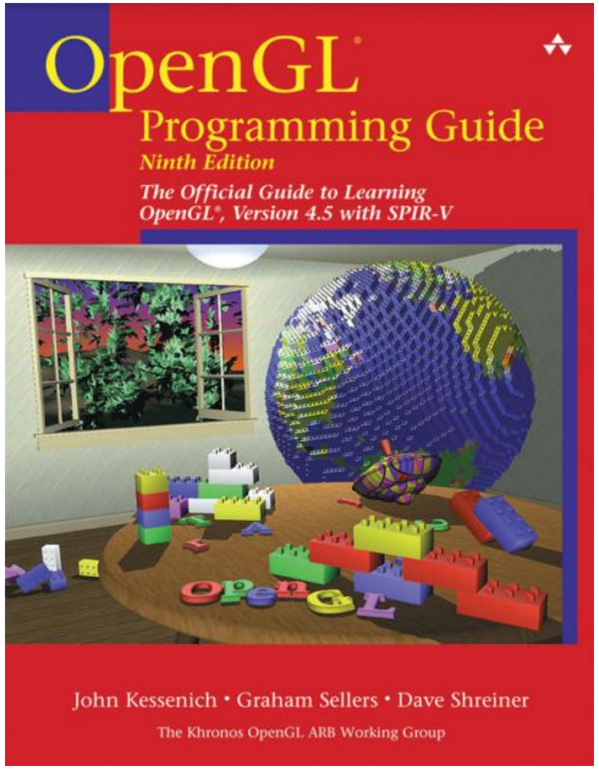
- Ability to read, write, and debug C++ programs
- Understanding of very basic data structures
- No serious software design required
- Vector geometry (dot/cross products)
- Linear algebra (basic matrices in 2-4D)
- Basic calculus (simple derivatives)

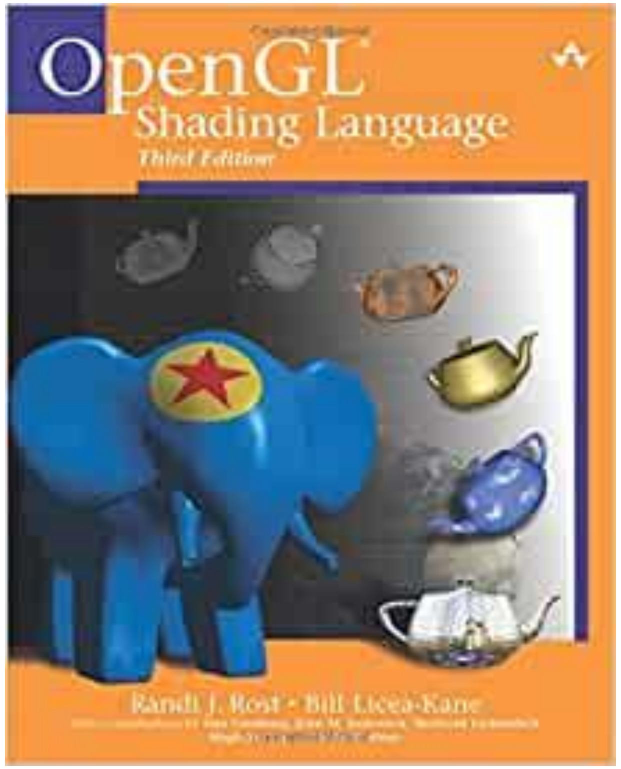
Workload

- 6-7 programming assignments
- 1 final project & optional representation
- Submit experimental report to elearning.ecnu.edu.cn
  - Purpose
  - Content and setup
  - Environment
  - Process and analysis
  - Summary of experimental results
- Template and guidance document will be available on both tencent meeting and elearning.ecnu.edu.cn

Textbook


Fundamentals of Computer Graphics
FOURTH EDITION
Steve Marschner
Peter Shirley
with
Michael Ashikhmin
Michael Gleicher
Naty Hoffman
Garrett Johnson
Tamara Munzner
Erik Reinhard
William B. Thompson
Peter Willemsen
Brian Wyvill
CRC Press
AN A K PETERS BOOK


OpenGL
Programming Guide
Ninth Edition
The Official Guide to Learning OpenGL®, Version 4.5 with SPIR-V
John Kessenich • Graham Sellers • Dave Shreiner
The Khronos OpenGL ARB Working Group


OpenGL
Shading Language
Third Edition
Randi J. Rost • Bill Licea-Kane

Q&A

实验1:图形API基本操作

Programming with Graphics API

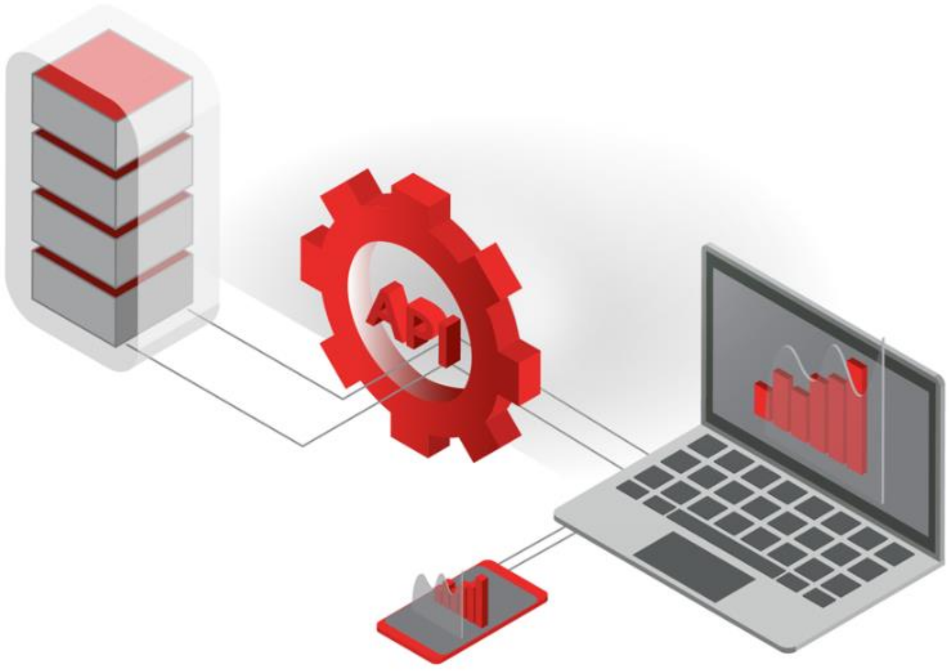
华东师范大学计算机科学与技术学院

李晨 副研究员

cli@cs.ecnu.edu.cn

Application Programming Interface(API)

- An Application Programming Interface(API) is a set of commands, functions and protocols which programmers can use when building a software.
- It allows the programmers to use predefined functions to interact with systems, instead of writing them from scratch.



API Example

- Think of an API like a menu in a restaurant. The menu provides a list of dishes you can order, along with a description of each dish. When you specify what menu items you want, the restaurant’s kitchen does the work and provides you with some finished dishes. You don’t know exactly how the restaurant prepares that food, and you don’t really need to.

Characteristics of Good API

Characteristics of Good API

- Easy to learn
- Easy to use, even without documentation
- Hard to misuse
- Easy to read and maintain code that uses it
- Sufficiently powerful to satisfy requirements
- Easy to evolve
- Appropriate to the audience

Windows API

- Operating systems (Windows, MacOS, Linux) usually provide a lot of APIs
- You basically program with APIs for most of real applications
- Windows Application Programming Interface
  - Base Services
  - Advanced Services
  - **GUI: Graphical User Interface**
  - **GDI: Graphics Device Interface**
  - Common Dialog Box Library
  - Common Control Library
  - Windows Shell
  - Network Services

Windows GUI

- C++ : Qt, **MFC**, WTL, wxWidgets, DirectUI, Htmlayout
- C# : WinForm, WPF
- Java : AWT, Swing
- Pascal: Delphi
- Go: walk, electron
- Web: Webkit, Chromium

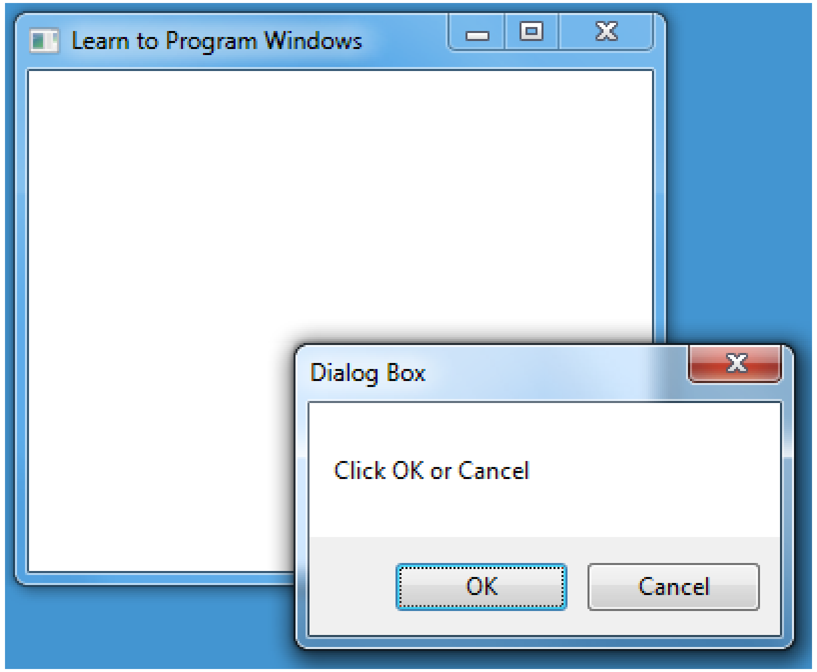
Microsoft Foundation Classes(MFC)

- There are about 200 MFC classes (more than 2000 API functions)
- They provide a framework upon which to build Windows applications. They encapsulate most of the Win32 API in a set of logically organized classes.
- They offer the convenience of reusable code. Many tasks common to all Windows apps are provided by MFC.
- They produce smaller executables.
- They can lead to faster program development.
- MFC programs must be written in C++ and require the use of classes.

Windows GDI

- Purpose
  - The Microsoft Windows graphics device interface (GDI) enables applications to use graphics and formatted text on both the video display and the printer. Windows-based applications do not access the graphics hardware directly. Instead, GDI interacts with device drivers on behalf of applications.
- Where applicable
  - GDI can be used in all Windows-based applications.
- Developer audience
  - This API is designed for use by C/C++ programmers. Familiarity with the Windows message-driven architecture is required.

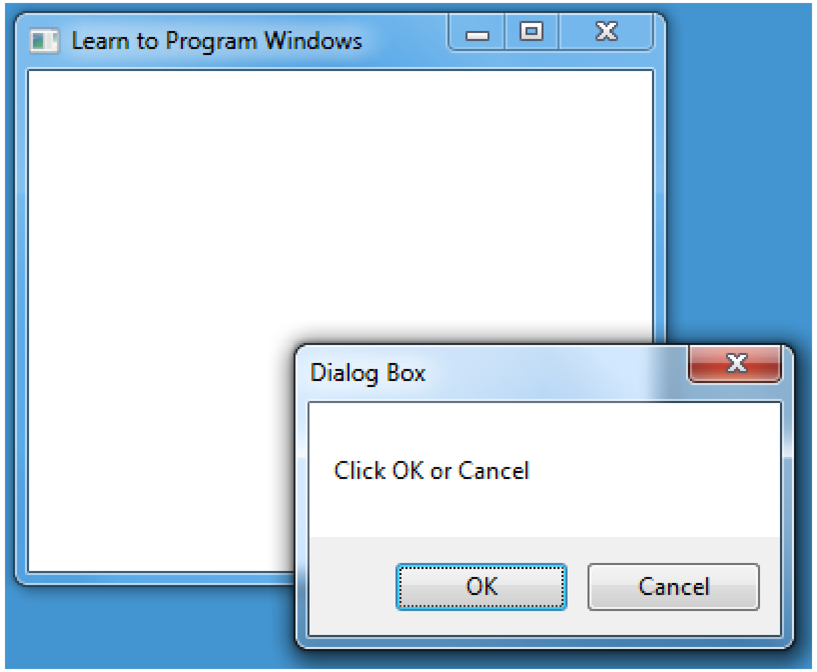
What is a Window?



Click OK or Cancel

What is a Window?

- When you think window, do not simply think application window. Instead, think of a window as a programming construct that:
  - Occupies a certain portion of the screen.
  - May or may not be visible at a given moment.
  - Knows how to draw itself.
  - Responds to events from the user or the operating system.



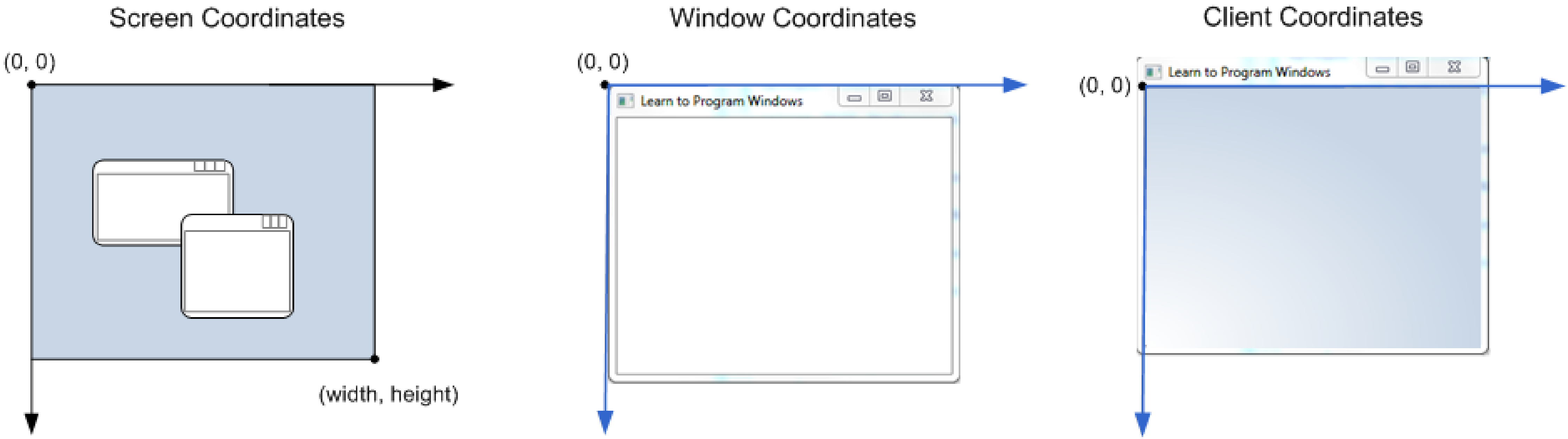
Click OK or Cancel

Window Handle

- It is just a number that the operating system uses to identify an object.
- You can picture Windows as having a big table of all the windows that have been created.
- It uses this table to look up windows by their handles.

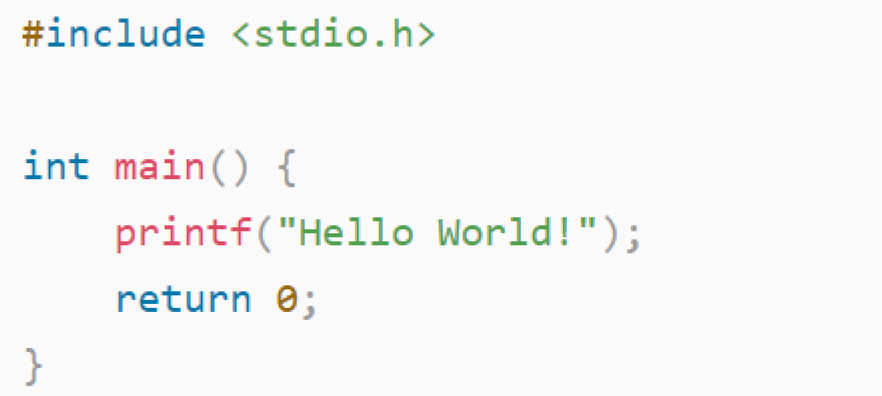
- Data type: `HWND`
- `CreateWindow` and `CreateWindowEx`.

Screen and Window Coordinates

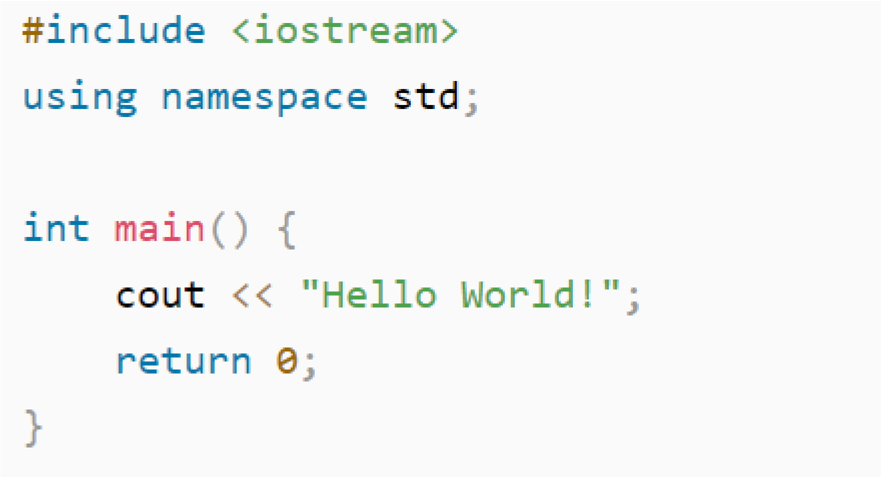


The Application Entry Point

- `int WINAPI wWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PWSTR pCmdLine, int nCmdShow)`
  - `hInstance` is something called a "handle to an instance" or "handle to a module." The operating system uses this value to identify the executable (EXE) when it is loaded in memory.
  - `hPrevInstance` has no meaning. It was used in 16-bit Windows, but is now always zero.
  - `pCmdLine` contains the command-line arguments as a Unicode string.
  - `nCmdShow` is a flag that says whether the main application window will be minimized, maximized, or shown normally.



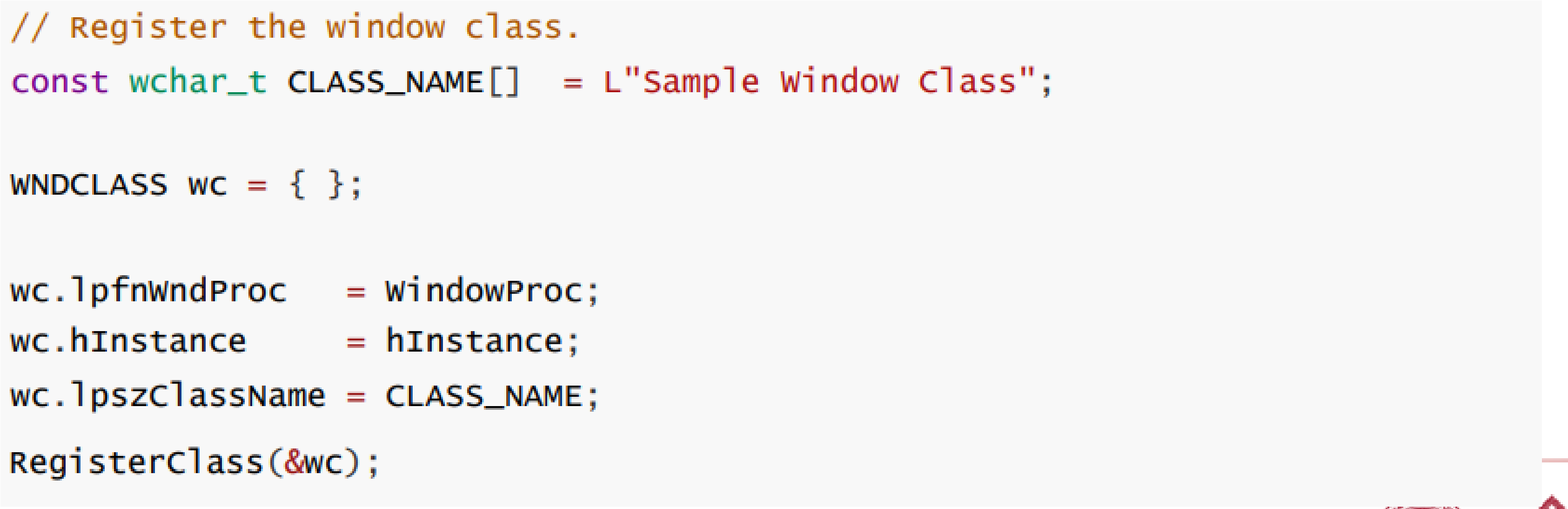
```
#include <stdio.h>

int main() {
    printf("Hello World!");
    return 0;
}
```



```
#include <iostream>
using namespace std;

int main() {
    cout << "Hello World!";
    return 0;
}
```

Creating a Window

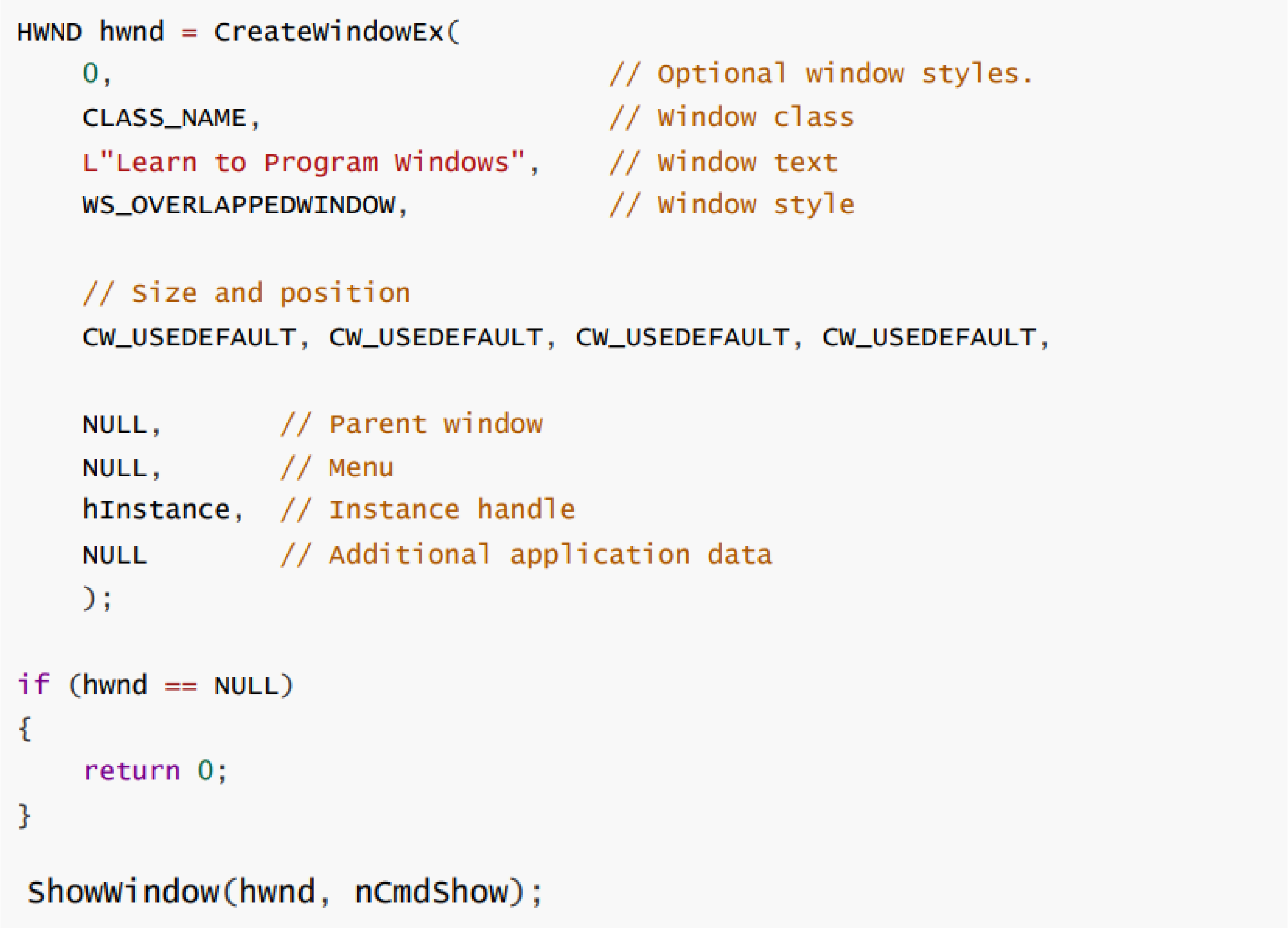
- A window class is not a "class" in the C++ sense. Rather, it is a data structure used internally by the operating system. Window classes are registered with the system at run time. To register a new window class, start by filling in a `WNDCLASS` structure:



```
// Register the window class.
const wchar_t CLASS_NAME[]  = L"Sample Window Class";

WNDCLASS wc = { };

wc.lpfnWndProc   = WindowProc;
wc.hInstance     = hInstance;
wc.lpszClassName = CLASS_NAME;

RegisterClass(&wc);
```

Creating a Window

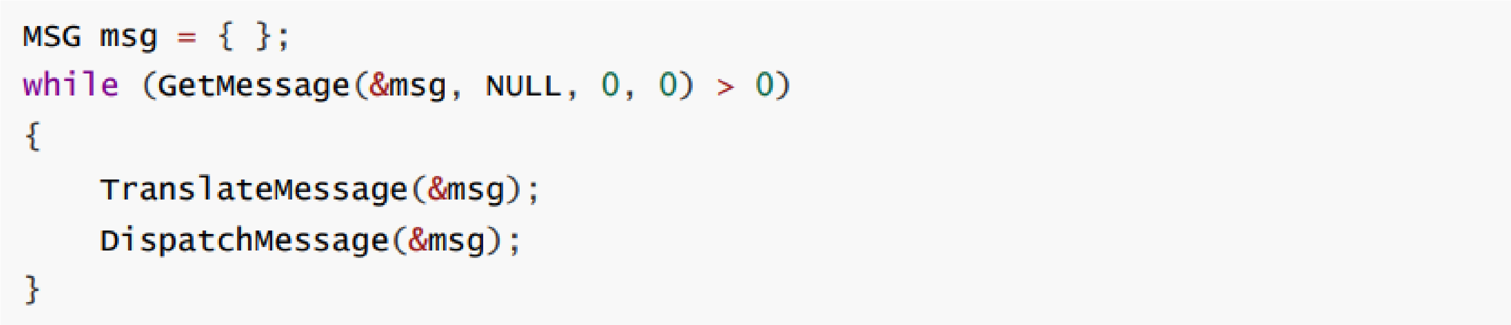
- `CreateWindowEx` returns a handle to the new window, or zero if the function fails.
- To show the window, pass the window handle to the `ShowWindow` function.



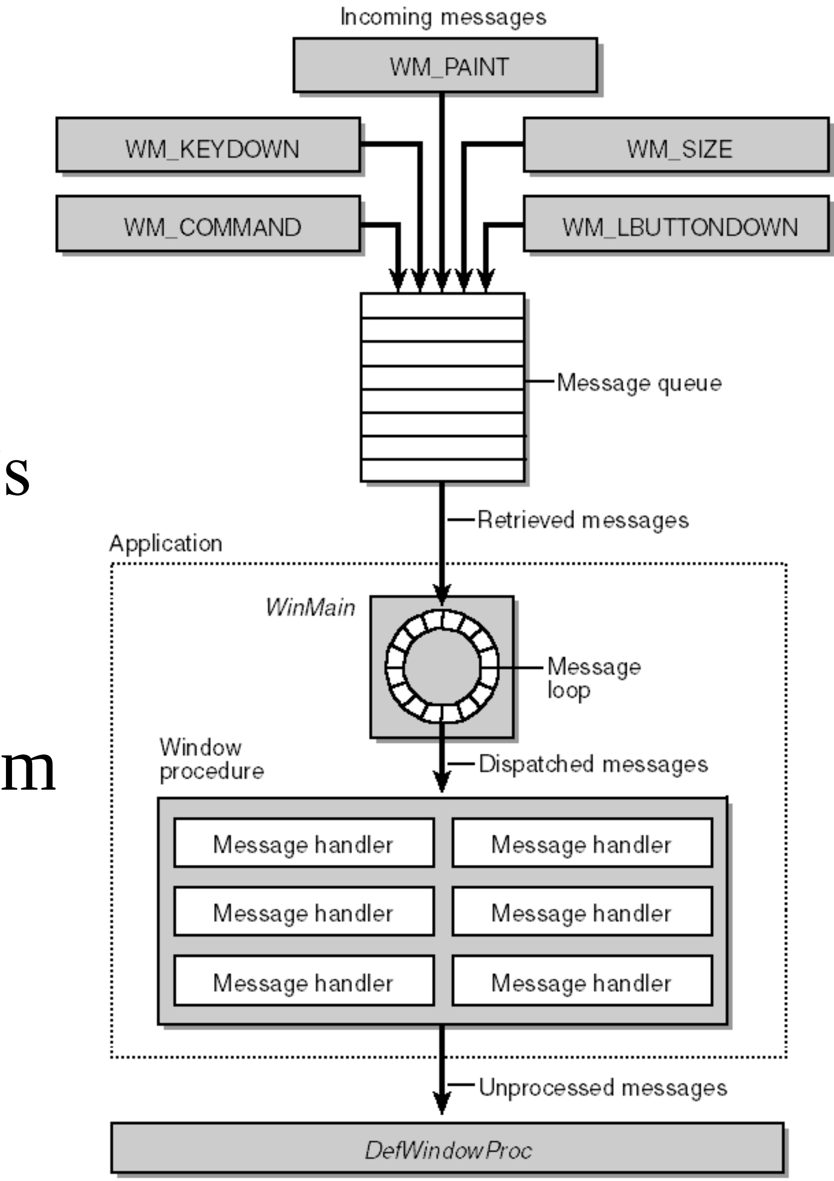
```
HWND hwnd = CreateWindowEx(
    0,                              // Optional window styles.
    CLASS_NAME,                     // Window class
    L"Learn to Program Windows",    // Window text
    WS_OVERLAPPEDWINDOW,            // Window style

    // Size and position
    CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,

    NULL,       // Parent window
    NULL,       // Menu
    hInstance,  // Instance handle
    NULL        // Additional application data
    );

if (hwnd == NULL)
{
    return 0;
}

ShowWindow(hwnd, nCmdShow);
```

Window Messages

- OS handles all device related messages
- There is a message queue outside our application's view
- We only handle things we care about
- This is quite different from terminal-based program

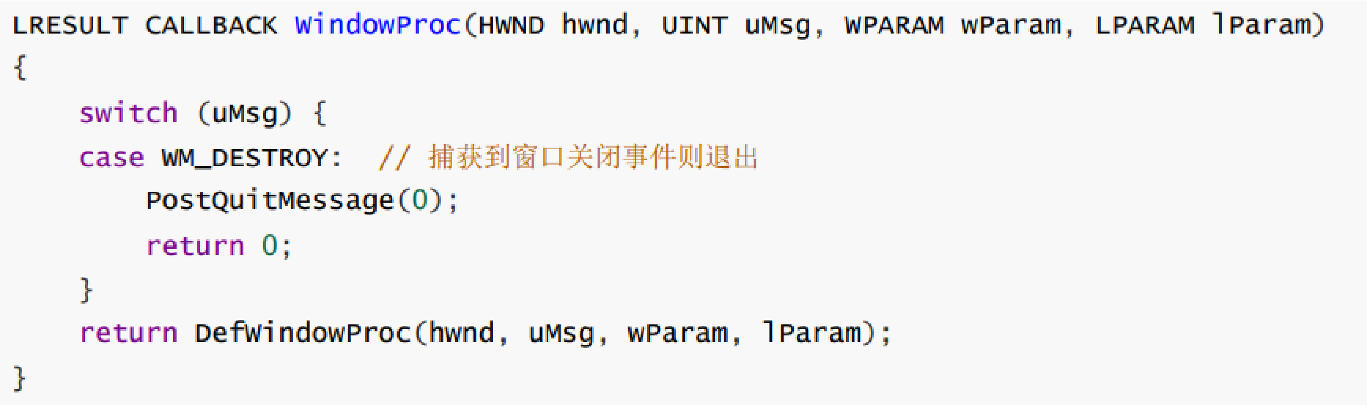


```
MSG msg = { };
while (GetMessage(&msg, NULL, 0, 0) > 0)
{
    TranslateMessage(&msg);
    DispatchMessage(&msg);
}
```



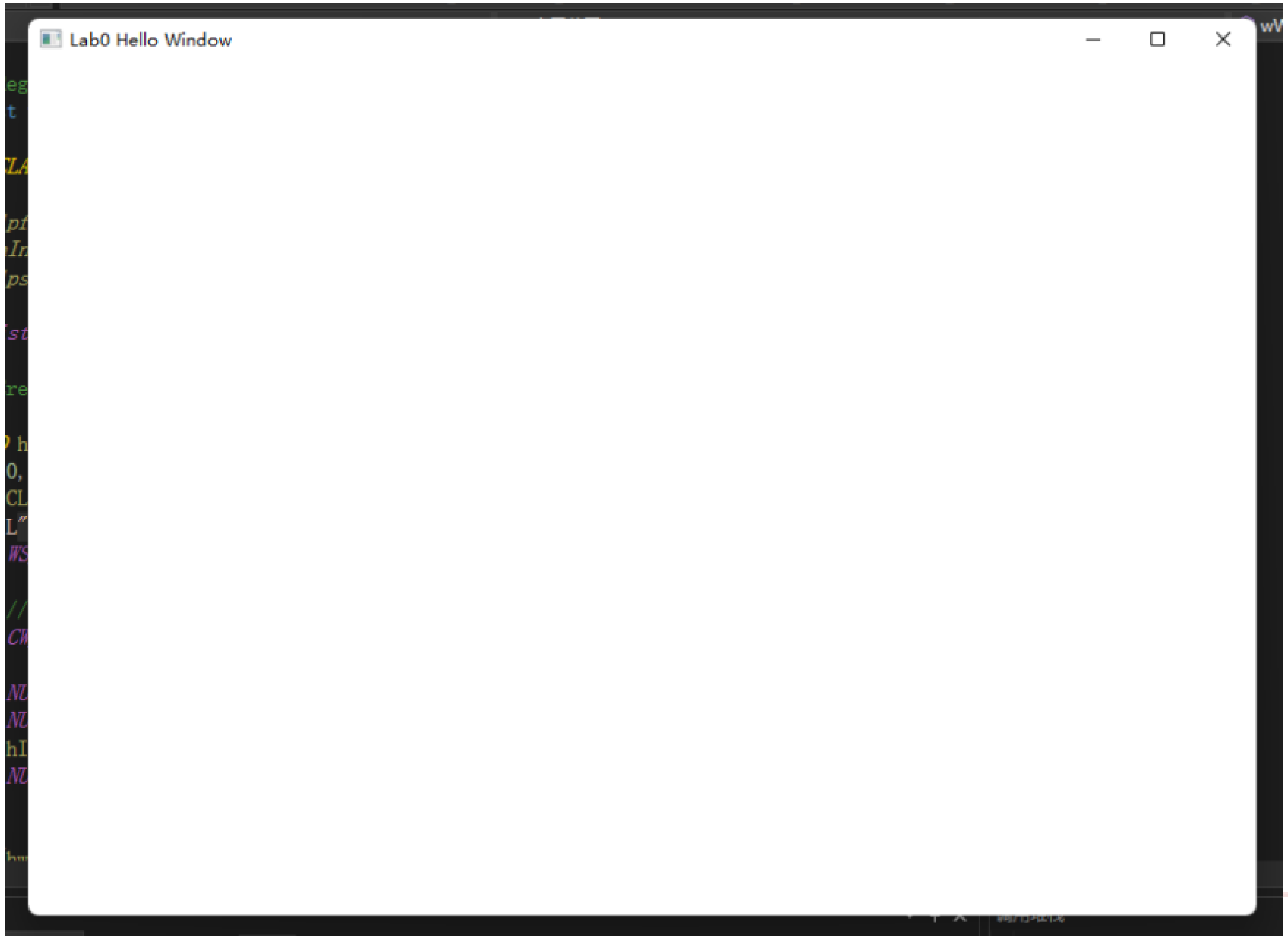
Windows Procedure

- hwnd is a handle to the window.
- uMsg is the message code; for example, the WM_SIZE message indicates the window was resized.
- wParam and lParam contain additional data that pertains to the message. The exact meaning depends on the message code.

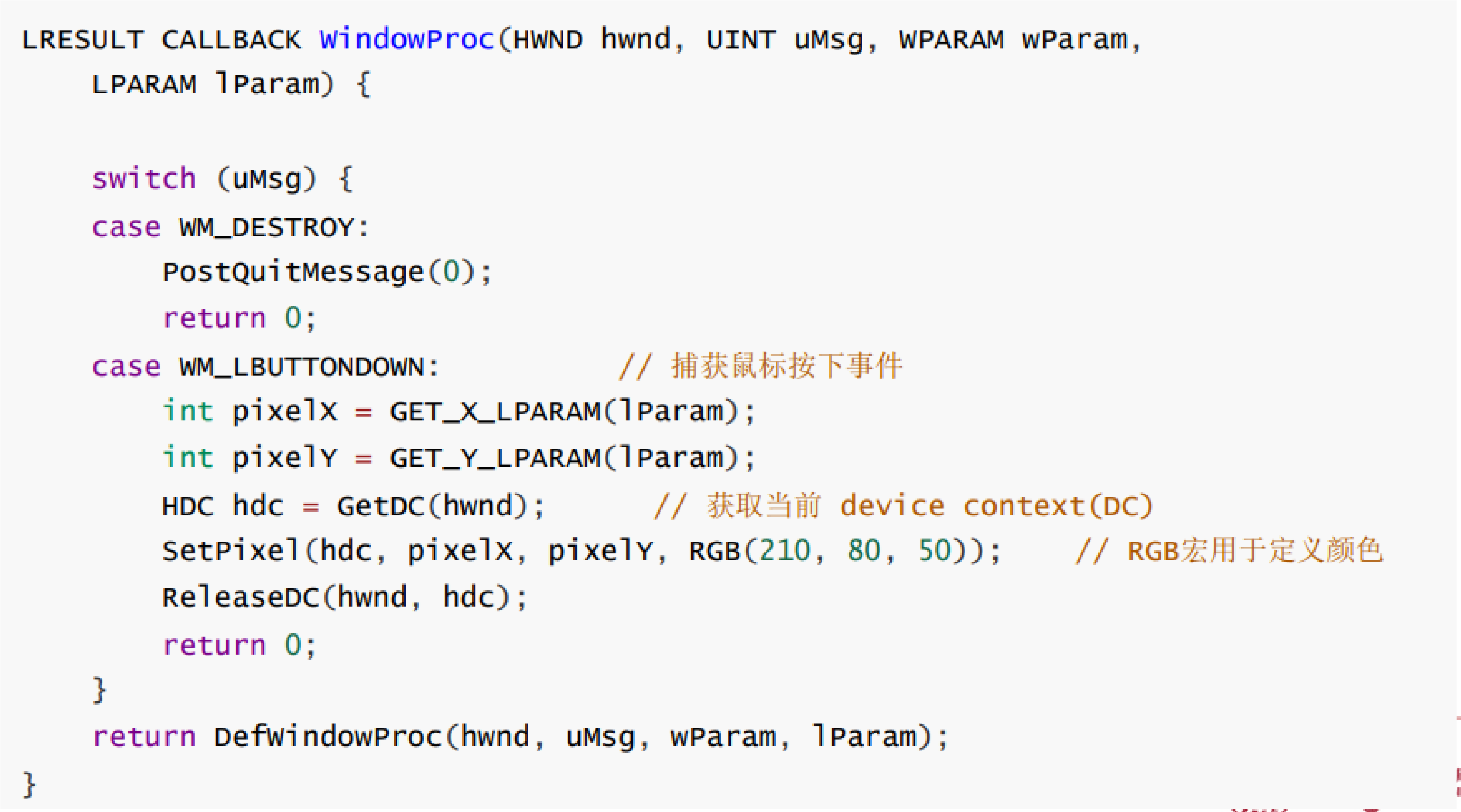


```
LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    switch (uMsg) {
    case WM_DESTROY:  // 捕获到窗口关闭事件则退出
        PostQuitMessage(0);
        return 0;
    }
    return DefWindowProc(hwnd, uMsg, wParam, lParam);
}
```

Your First Window



Assignment: Simple Drawing Board



```
LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam,
    LPARAM lParam) {

    switch (uMsg) {
    case WM_DESTROY:
        PostQuitMessage(0);
        return 0;
    case WM_LBUTTONDOWN:            // 捕获鼠标按下事件
        int pixelX = GET_X_LPARAM(lParam);
        int pixelY = GET_Y_LPARAM(lParam);
        HDC hdc = GetDC(hwnd);        // 获取当前 device context(DC)
        SetPixel(hdc, pixelX, pixelY, RGB(210, 80, 50));    // RGB宏用于定义颜色
        ReleaseDC(hwnd, hdc);
        return 0;
    }
    return DefWindowProc(hwnd, uMsg, wParam, lParam);
}
```

Assignment: Simple Drawing Board

- `WM_LBUTTONUP` : Posted when the user releases the left mouse button while the cursor is in the client area of a window.
- `WM_LBUTTONDOWN`: Posted when the user presses the left mouse button while the cursor is in the client area of a window.
- `WM_MOUNSEMOVE`: Posted to a window when the cursor moves.
- Set `is_draw` to true when the left mouse is pressed, and to false when the left mouse is released, and capture `WM_MOUNSEMOVE` Simultaneously.
- Call `SetPixel` at the mouse position for `WM_MOUNSEMOVE` message.

Assignment: Simple Drawing Board

- 实验编号：1
- 实验名称：图形API基本操作
- 实验目的：实现操作系统下图形API的基本绘制功能。
- 实验环境：
  - Windows 10
  - Visual studio X

Reference

- https://docs.microsoft.com/en-us/windows/win32/learnwin32/creating-a-window
- https://docs.microsoft.com/en-us/cpp/windows/walkthrough-creating-windows-desktop-applications-cpp?view=msvc-160&viewFallbackFrom=vs-2019
- https://docs.microsoft.com/en-us/windows/win32/api/wingdi/nf-wingdi-setpixel