Installing OpenGL and Running the Book's Code

Our programs have all been tested to run on a Microsoft Visual Studio Community (MSVS) 2019 IDE running on Windows 10 on a 64-bit PC with a graphics card supporting OpenGL 4.3. They were either developed in this environment as well or in an earlier version of MSVS and then retargeted.

Described below is how to set the development environment up on MSVS Commumity 2019/64-bit Windows 10. These instructions should be applicable to other versions of Windows and Visual Studio (including the newest version MSVS Community 2022 at the time of writing) and to a 32-bit machine as well; our programs, possibly with minor changes, should all run on a Linux or Mac OS environment. Keep in mind that, though a graphics card supporting at least OpenGL 4.3 is ideal, a lower version of OpenGL might be fine too – see a note at the end of this guide.

- Install Visual Studio

  1. Download Microsoft Visual Studio Community 2019 from `https://my.visualstudio.com/Downloads?q=visual%20studio%202019&wt.mc_id=o~msft~vscom~older-downloads` and follow the installation steps.

- Install Helper Libraries

  1. Create a folder called `OpenGLwrappers` in the `C:` drive, so this folder is `C:\OpenGLwrappers`. This particular name and location is so that all our programs in `ExperimenterSource` run out of the box.

     The version numbers below of the libraries we ask you to download may not be the latest but they are the ones we have installed in our environment. There are sometimes little things that differ with versions. To be sure, we have not yet heard of a problem in upgrading to a newer version. Still, to avoid any issues at all while you are learning we suggest you install the exact versions below.

  2. FreeGLUT: Download and unzip the file `http://files.transmissionzero.co.uk/software/development/GLUT/older/freeglut-MSVC-2.8.1-1.mp.zip` and save the folder `freeglut-MSVC-2.8.1-1.mp` in `OpenGLwrappers`.
  3. GLEW: Download and unzip the file `https://sourceforge.net/projects/glew/files/glew/1.10.0/glew-1.10.0-win32.zip/download` and save the folder `glew-1.10.0-win32` in `OpenGLwrappers`.
  4. GLM: Download and unzip the file `https://github.com/g-truc/glm/releases/download/0.9.7.5/glm-0.9.7.5.zip` and save the folder `glm-0.9.7.5` in `OpenGLwrappers`.
  5. Copy `freeglut.dll` from `C:\OpenGLwrappers\freeglut-MSVC-2.8.1-1.mp\freeglut\bin` to `C:\Windows\SysWOW64`.

6. Copy `glew32.dll` from `C:\OpenGLwrappers\glew-1.10.0-win32\glew-1.10.0\bin\Release\Win32` to `C:\Windows\SysWOW64`.
7. Check if `glu32.dll` is already in `C:\Windows\SysWOW64`. Normally, it should be. If not, search resources on the web to reinstall it there.

- Test the Environment

    1. Download the files `Experimenter.pdf` and `ExperimenterSource.zip` from the book's website `http://www.sumantaguha.com`. Unzip the latter file and place `ExperimenterSource` in the *same* folder as `Experimenter.pdf`.
    2. Open `Experimenter.pdf` with (preferably) Adobe Reader. Go to Chapter 2 and click Windows Project just above Experiment 2.1 to bring up the Square project in MSVS. (Make sure you have MSVS 2019 as the default app associated with the `.vcxproj` extension.)
    3. On the MSVS tool bar click Build → Build Solution to compile the project, which should show 1 succeeded and 0 failed in the Output pane at the bottom.
    4. On the tool bar click Debug → Start Debugging to run. This should bring up two windows: an all black C++ window and the OpenGL window with a black square drawn on a white background (Figure 1 is a screenshot of the latter window). Do you see these two windows? Great! You are all set.

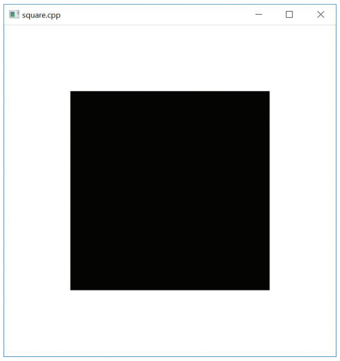

Figure 1: Screenshot of `square.cpp`, in particular, the OpenGL window.

Of course, you can run the programs directly in `ExperimenterSource` by going to the program folder and opening either the `.vcxproj` or `.sln` file, without going through `Experimenter.pdf`. However, `Experimenter.pdf` has the programs in the same sequence that they appear in the book so the two work nicely together.

Next, we'll show you how to make your own OpenGL project as surely you will be wanting to soon enough. First, though, we recommend creating an OpenGL project template which can then be used for all future projects.

- Create and Use an OpenGL Project Template

    1. Open Visual Studio 2019 from the Start Menu. Click "Create a new project" and in the next page search for "C++" in the templates search box and select C++ Console App. Click Next. In the "Configure your new project" page enter `OpenGLProjectTemplate` for Project name and for Location any convenient folder. Check the box "Place solution and project in the same directory". Click Create.
    2. The MSVS page of `OpenGLProjectTemplate` comes up. On the tool bar click View → Solution Explorer to open the Solution Explorer pane. Under the Source Files item you will find `OpenGLProjectTemplates.cpp`. Double click it to open it in another pane.

3. Open the file `square.cpp` in the folder `ExperimenterSource\Chapter2\Square` in a text editor and copy its contents into `OpenGLProjectTemplate.cpp` overwriting the existing contents. Ignore the multiple red lines that appear for now.
4. On the tool bar of the MSVS page click Project → OpenGLProjectTemplate Properties to bring up the Property Pages. Select All Configurations from the dropdown menu to the right of "Configuration:".
5. Expand Configuration Properties and click Configuration Properties → C/C++ → General → Additional Include Directories → ⟨Edit. . .⟩ (in the drop-down menu) to open the Additional Include Directories dialog box. Click the New Line icon successively to create a new line to add each of the following folders (order doesn't matter; you can click the box with ". . ." at the right of the empty new line to navigate to the folder):
   (a) `C:\OpenGLwrappers\freeglut-MSVC-2.8.1-1.mp\freeglut\include`
   (b) `C:\OpenGLwrappers\glew-1.10.0-win32\glew-1.10.0\include`
   (c) `C:\OpenGLwrappers\glm-0.9.7.5\glm`

   Click OK to confirm.
6. Click Configuration Properties → Linker → General → Additional Library Directories → ⟨Edit. . .⟩ (in the drop-down menu) to open the Additional Library Directories dialog box. Click the New Line icon successively to create a new line to add each of the following folders (order doesn't matter; you can click the box with ". . ." at the right of the empty new line to navigate to the folder):
   (a) `C:\OpenGLwrappers\freeglut-MSVC-2.8.1-1.mp\freeglut\lib`
   (b) `C:\OpenGLwrappers\glew-1.10.0-win32\glew-1.10.0\lib\Release\Win32`

   Click OK to confirm.
7. Click Configuration Properties → Linker → Input → Additional Dependencies → ⟨Edit. . .⟩ (in the drop-down menu) to open the Additional Dependencies dialog box. Manually enter the following file names one per line in the window at the top (order doesn't matter):
   (a) `glew32.lib`
   (b) `opengl32.lib`

   Click OK to confirm.
8. Click OK to dismiss the Property Pages. All the red lines in the source file `OpenGLProjectTemplates.cpp` should now be gone. Again, Build Solution and Start Debugging and you should see a black C++ window and an OpenGL window as in Figure 1.

9. Expand Source Files in the Solution Explorer pane, right click `OpenGLProjectTemplates.cpp` and Remove → Delete as it's no longer needed.
10. On the MSVS toolbar click File → Save All. Next click Project → Export Template. In the Export Template Wizard choose the Project template radio button and click Next. Name the template `OpenGLProjectTemplate` and check the box "Automatically import the template into Visual Studio" and uncheck the box "Display an explorer window on the output files folder". Click Finish. The template `OpenGLProjectTemplate` has been created and you can, in fact, now delete the project `OpenGLProjectTemplate` from the folder you created it in.
11. To use the template, open Visual Studio 2019 from the Start Menu to bring up the MSVS Start Page. Click Create a new project and search for and select the template `OpenGLProjectTemplate`. Click Next. Name the project and select an appropriate folder for its location and click Create. The new project comes up. Right click on Source Files in the Solution Explorer to add a New Item. Choose C++ File for the type of this item, give a Name, and Add. A `.cpp` file pops up in a pane in which you can now write C++/OpenGL code.

OpenGL support: Check what level of OpenGL your graphics card supports by running an OpenGL extensions viewer (you can download one from `http://www.realtech-vr.com/glview/download.php`). If it doesn't support OpenGL 4.3 then a book program may compile but not run as the system is unable to provide the OpenGL 4.3 rendering context asked by the command `glutInitContextVersion(4, 3)` in the main routine. What you might do in this case is replace `glutInitContextVersion(4, 3)` in main with `glutInitContextVersion(3, 3)`, or even `glutInitContextVersion(2, 1)`, instead. Of course, then, programs using later-generation calls will not run, but you should be fine early on in the book.