ICS 161 – Winter, 2016 – Homework 1

Follow the steps below. Start in your Wednesday, Jan. 6 lab session, and complete the steps on your own time. As specified below, take several screen shots that show your progress and paste them (in order) into a single Word document. Upload that Word doc to the EEE Dropbox "ICS161-Hw1" before 11:55pm on Wednesday, Jan. 13.

- You can take a screen shot of the active window by holding down the Alt key and pressing the PrtScn key. This puts an image in the Windows clipboard, which you can paste into a Word document with Ctrl-V. Sometimes Alt-PrtScn does not work. If not, use plain PrtScn instead with no maximized windows, and scale the copied image to fit on a single Word page. With plain PrtScn you may need to make sure that the focus is on the program you are running, by clicking on the command prompt window.
- The desktops in the COGS lab are not shared. Files you store on the Desktop or in My Documents will not be available on other computers, and may disappear from the computer you created them on. You should always copy your files to a USB stick or a network drive.

We'll more or less follow the SDL lessons at http://www.willusher.io/, starting with several Lesson 0s.

0 + 0 + 0 = A good start

1. Read Lesson 0, Setting Up SDL at http://www.willusher.io/sdl2%20tutorials/2013/08/15/lesson-0-setting-up-sdl/ before proceeding. The next few steps of this homework will guide you through "The first step" in Lesson 0.

2. Go to http://www.libsdl.org/download-2.0.php and download the source code in SDL-2.0.4.zip [was -2.0.3.zip] onto the Desktop. Extract the zip to C:\Users\xxx\Desktop\SDL. (You can of course use a different location and folder name, but the instructions below assume Desktop\SDL.)

3. Some COGS lab computers have Visual Studio 2013 installed on them; others may only have 2012. You can find out by looking at the Start menu. The instructions below assume 2013, but you can use 2012 with only a few modifications. The first time you start Visual Studio 2012 or 2013, you will be asked to provide some defaults.

4. In Windows Explorer, navigate to C:\Users\xxx\Desktop\SDL\SDL2-2.0.4\VisualC. Unfortunately, the 2.0.4 release of SDL2 is missing the SDL_VS2013.sln solution file for Visual Studio 2013. Instead, double click on SDL_VS2008.sln, which will start up Visual Studio 2013. Click OK for the One-way upgrade. Ideally 24 projects will be "migrated" to VS 2013. Take a brief glance at the Migration Report, and then close it. The first time you start Visual Studio 2013, you may be asked to provide some defaults.

5. In Windows Explorer, note the files in the SDL folder under the VisualC folder. There should be four or five files.

   In Visual Studio, make sure SDL2 is highlighted (selected) in the Solution Explorer pane. Note that Debug and Win32 are selected in drop down boxes below the menu. From the menu, choose Build | Build SDL2. After a minute or two you should see Build: 1 succeeded, 0 failed. **Take a screen shot showing the successful build.**

   Note: If you are running a Windows 8 or Windows 8.1 OS, your build will likely fail with “Cannot open include file: 'dxsdkver.h': No such file or directory”. This can be fixed by expanding the SDL2 project in the Solution Explorer, opening the SDL_xaudio2.c file, and commenting out line 68 which reads “#include <dxsdkver.h>” using //.

6. Note that new folders and files have been created in the SDL subfolder. Find SDL2.dll somewhere in the SDL2-2.0.4 folder.

7. Build SDL2main and SDL2test.

8. Experiment with some of the projects under “tests”. These are finicky and don’t seem to be set up correctly. A procedure that usually works is to highlight the project name (e.g. testplatform), right click on the name, and choose Build; then right click again and choose Debug | Start new instance. A dialog box will tell you that one or more other projects (specifically SDL2) are out of date – this is incorrect, so reply “No” to “Would you like to build it?” Note that in some cases the interesting output goes to the Visual Studio Output pane. Feel free to ignore the “Cannot find or open the PDB file” warnings.

9. Now you are ready for the second Lesson 0, Visual Studio at http://www.willusher.io/sdl2%20tutorials/2013/08/15/lesson-0-visual-studio/. Before you start, close the SDL2 solution in Visual Studio. Create a new subfolder in Desktop\SDL called MyProjects.

10. In Visual Studio, choose File | New | Project… | Visual C++ | Empty Project. Follow Lesson 0, but use C:\Users\xxx\Desktop\SDL\MyProjects\ for the Location and “SDL_Tutorial” for the project name.

11. Follow “Setting the Include Path” – access the dialog with Project | Properties. A challenge left for you is to find the path to the include directory (hint: the include directory has a lot of .h files in it).

12. In “Setting the Library Path” you will not see a “lib” folder, for reasons that are explained several steps below. Instead, find a folder specific to the Debug configuration which contains SDL2.lib and SDL2main.lib. (Since we are setting the properties for All Configurations, this will cause a problem if we want to build with Release in the future. Let’s keep that in mind but not worry about it now.)

13. Continue with “Adding the Library Dependencies” and “Selecting the Subsystem”.

14. In “The Test Program” you can create a new source code file by right clicking on Source Files in the Solution Explorer pane, and selecting Add | New item… | C++ file with any name you choose that’s better than “Source.cpp”. Copy the code from the tutorial into the new .cpp file. Build and run the project (you can use the green play arrow labeled Local Windows Debugger), and make sure you don’t have any errors. You will get a “The program can’t start because SDL2.dl is missing” error – but the .dll isn’t missing, it just isn’t in a place where Windows looks. Copy SDL2.dll from its location deep within SDL2-2.0.4\ to C:\Users\xxx\Desktop\SDL\MyProjects\SDL_Tutorial\Debug (so that it is in the same folder as the .exe file) and try again.

15. Follow the instructions for Exporting a Template Project. Close the solution.

16. On to Postscript 0 at http://www.willusher.io/sdl2%20tutorials/2014/06/16/postscript-0-properly-finding-resource-paths/. You can create a .h file in Visual Studio with File | New | File… | Visual C++ | Header File. Copy the block of code in “Getting the Resource Path” into your new .h file, and save it as res_path.h in C:\Users\xxx\Desktop\SDL\SDL2-2.0.4\include.

17. The instructions in “Using the Resource Path Lookup” about adding a directory to our include path aren’t really useful, since we’ve put res_path.h in the standard include directory.

18. Create a new project in MyProjects based on the SDL_Template template, with a single C++ file containing the code given in the Postscript’s last code block. Exiting and restarting Visual Studio may be necessary for the SDL_Template to show up.

19. You’ll find that the .cpp file has the same name as in the template – it’s probably a good idea to rename it to something appropriate to this project, which you can do inside the Solution Explorer pane. If you get an error message regarding SDL2.dll, copy it into the new Debug folder.

20. You’ll see that writing to std::cout causes a command prompt window to appear, which then quickly disappears. This is annoying. Add the following line of code after the std::cout << line: `std::cin.ignore();`. Build and rerun and observe the output at your leisure. Press Enter to continue.

21. The only problem is that the directory structure getResourcePath() is assuming is not the one Visual Studio creates. Visual Studio's directory structure looks like this (I've added a res\ folder for resources which isn't created by Visual Studio but which you will add later):

```
    MyProjects
        Lesson1
            Debug (not bin – the executable lives here)
        Lesson2
            Debug
        res
            Lesson1
            Lesson2
```

    So, we need to modify getResourcePath() in res_path.h. Delete or comment out the two lines following the "We replace the" comment, and insert these lines:

```
    size_t pos = baseRes.rfind("Debug");
    if (pos == std::string::npos)  // no "Debug", assume "Release"
        pos = baseRes.rfind("Release");
    size_t prevSlash = baseRes.rfind(PATH_SEP, pos-2); // find slash
before project name
    baseRes = baseRes.substr(0, prevSlash) + PATH_SEP + "res" +
PATH_SEP;
```

    Save res_path.h, rebuild the project, and carefully examine the output. Before you press Enter, **take a screen shot showing the output in the command prompt window.**

Lesson 1 – Hello World

1. Lesson 1 is at http://www.willusher.io/sdl2%20tutorials/2013/08/17/lesson-1-hello-world/. Before you begin, create a Lesson1 project in MyProjects, based on SDL_Template. As before, you should rename the source code file to an appropriate name (e.g. Lesson1.cpp).

2. Download hello.bmp to the folder for Lesson 1 resources (which you will have to create).

3. Copy the code snippets from the web page into the program's main(). Visual Studio's Ctrl-K, Ctrl-D will reformat the file and correct indentation. Make sure you have an include for res_path.h. Make a copy of SDL2.dll if necessary. Run the program and **take a screen shot showing the output window.**

4. After you've run this program successfully, close the solution and head over to Postscript 1 at http://www.willusher.io/sdl2%20tutorials/2014/08/01/postscript-1-easy-cleanup/ .

5. The last two steps in this section require Visual Studio 2013 or a later version of VS 2012 than we have installed in the COGS lab. If you are at a computer without VS 2013, you can skip these steps, but still read the material closely, because you will have to make minor changes during the later lessons.

6. After you've read through the postscript, create a cleanup.h file in C:\Users\xxx\Desktop\SDL\SDL2-2.0.4\include which contains the provided code for "The Variadic Cleanup Function".

7. Return to Lesson1, make the changes to use cleanup() as specified in the Postscript, and run it with, ideally, no change in output.

Lesson 2 – Don't Put Everything in Main

1. Lesson 2 is at http://www.willusher.io/sdl2%20tutorials/2013/08/17/lesson-2-dont-put-everything-in-main/. Start off in the same way as Lesson 1, naming the project Lesson2 and the source code file Lesson2.cpp. Copy code from the lesson into your source code file. Create the appropriate folder for Lesson 2 resources. Correct a bug in the Lesson's code regarding the render variable. Copy SDL2.dll as needed (this is becoming tiresome, isn't it?).

2. If you are using VS2012, replace the calls to cleanup() with appropriate calls to SDL APIs, which you used in Lesson 1.

3. **Take a screen shot showing the output window**.

Lesson 3 – SDL Extension Libraries

1. Lesson 3 is at http://www.willusher.io/sdl2%20tutorials/2013/08/18/lesson-3-sdl-extension-libraries/ . As the lesson directs, go to http://www.libsdl.org/projects/SDL_image/ and download SDL_image_devel-2.0.1-VC.zip onto your Desktop. Extract the zip onto a Desktop folder, and then copy the contents of its include\ folder (one file!) over to C:\Users\xxx\Desktop\SDL\SDL2-2.0.4\include and the contents of its lib\x86\ folder over to C:\Users\xxx\Desktop\SDL\SDL2-2.0.4\VisualC\SDL\Win32\Debug.
2. The lesson directs Visual Studio users to "Include SDL_image.h and add SDL_image.lib to your linker dependencies." Let's do this by updating SDL_Template.
    a. In Visual Studio, create a new Project based on SDL_Template. Call it SDL_TemplateNew.
    b. Add a #include for SDL_image.h, and while you're at it, do the same for res_path.h and for cleanup.h.
    c. Update the project's properties under Linker, Input to list SDL2_image.lib.
    d. Make sure the project can build and run. Then use File | Export Template to create a template. You can call it SDL_Template and overwrite the existing SDL_Template.
    e. Hey, where do templates live, anyway? Take a look at My Documents\Visual Studio 2013\My Exported Templates.
    f. Close the Solution.
3. Continue with Lesson 3 by creating a new project and proceeding in the usual manner. Do the "optional" Initialize SDL_image part. The lesson doesn't make it super clear that three blocks of code from Lesson 2 need to be part of Lesson 3: the logSDLError function; the declaration, initialization, and nullptr checks for window and renderer in main(); and the call to cleanup(). Copy that code from Lesson2.cpp to Lesson3.cpp.
4. You'll need four DLL files in the same folder as Lesson3.exe: three from the zip you downloaded earlier, libpng16-16.dll, SDL2_image.dll, zlib1.dll, and good old SDL2.dll,. Yes, this is a pain and yes, we will address it soon.
5. **Take a screen shot showing the output window**.

That's it! Make sure to upload your doc file with the screen shots to EEE.