Monday, 13 May 2019

Installing OpenGL and FLTK Libraries

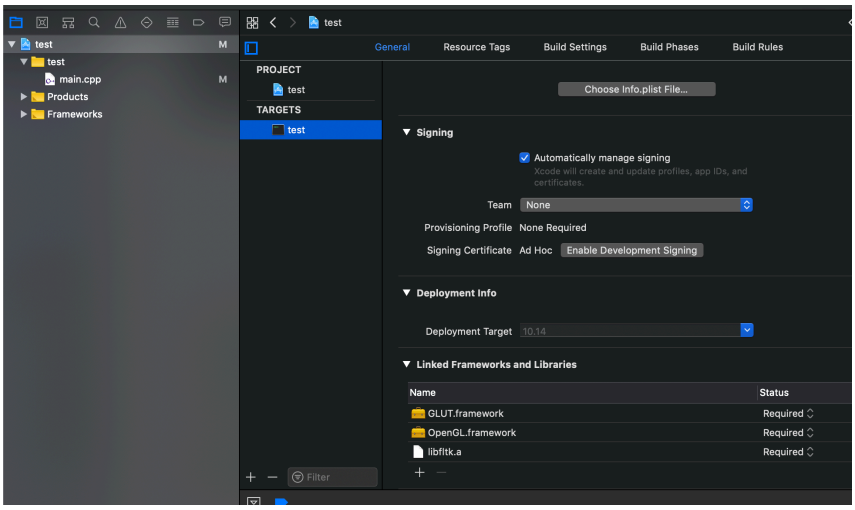
Installation on MacOS:

1. Update your OS to the latest version, Mojave 10.14.4 as per the writing of this document
2. Download Xcode from the App Store and install (you need 15GB of required space at least), make sure you include the *command line tool.* **This includes C/C++ compiler, GLUT and OpenGL frameworks**
3. After this is done, go to terminal and install Homebrew if you don't already have it:
   - ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
   - Check that you have installed brew correctly by typing: brew doctor
   - You should see the message "your system is ready to brew"
4. **Install FLTK** for Assignment 2 by typing the following to command line:

   ```
   brew install fltk
   ```

5. Check that FLTK is installed correctly in:
   - `/usr/local/lib` : libfltk.a exists
   - `/usr/local/include` : a folder named FL exists

Check if OpenGL and GLUT is installed correctly:



1. Open Xcode, **create** a new project of type "command line tools" under MacOS tab.

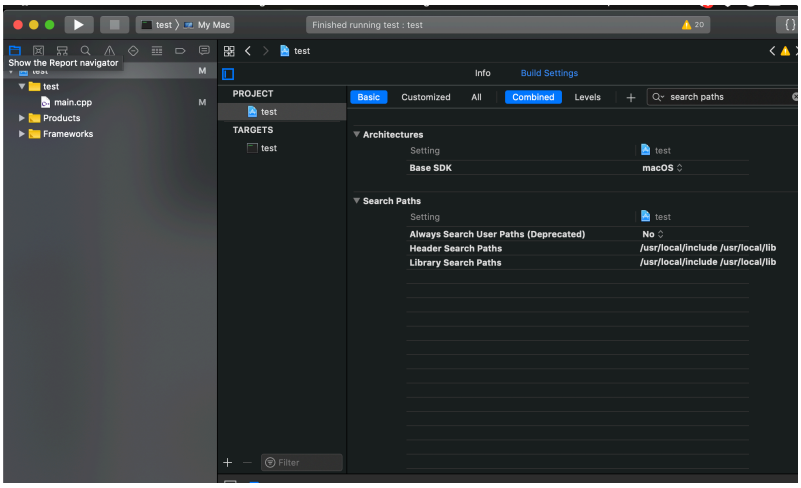
2. Choose a **location**, and save the project there

3. Xcode will bring you to your directory view.

-Set **framework** as

`OpenGL` and `GLUT` under the “general” tab for your target. See the screenshot above to see which items to select.

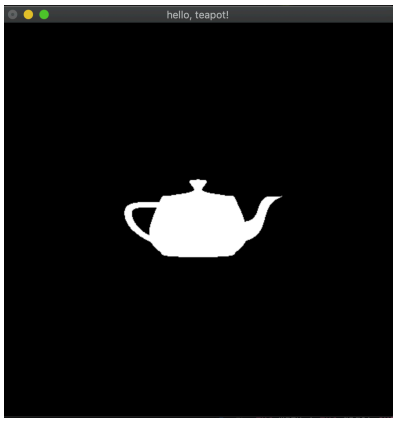
4. Add `libfltk.a` to the framework as well. You can do this by:
   - Click anywhere on desktop, the bar above should be “Finder”, click Go, then “Go to Folder…”
   - Type `/usr/local/lib`
   - Drag `libfltk.a` to the Linked Framework and Libraries area on Xcode
5. Go to build settings of your project (not target), search “search paths” and add the two search paths in both Header Search path and Library Search path : `/usr/local/lib` and `/usr/local/include`, each can be added using the + button. This tells Xcode where to find your libraries and header file for FLTK.



6. Click `main.cpp` file that is automatically created for you and paste the following code (overwrite whatever that was written there):

```
#include <OpenGL/gl.h>
#include <GLUT/glut.h>
#include <FL/Fl.H>
void display () {

    /* clear window */
    glClear(GL_COLOR_BUFFER_BIT);

    /* future matrix manipulations should affect the modelview matrix */
    glMatrixMode(GL_MODELVIEW);

    /* draw scene */
    glutSolidTeapot(.5);

    /* flush drawing routines to the window */
    glFlush();

}
```

```
void reshape ( int width, int height ) {

    /* define the viewport transformation */
    glViewport(0,0,width,height);

}

int main ( int argc, char * argv[] ) {

    unsigned eventCoordX = Fl::event_x();
    printf("Checking if FLTK works: %d\n", eventCoordX);

    /* initialize GLUT, using any commandline parameters passed to the
       program */
    glutInit(&argc,argv);

    /* setup the size, position, and display mode for new windows */
    glutInitWindowSize(500,500);
    glutInitWindowPosition(0,0);
    glutInitDisplayMode(GLUT_RGB);

    /* create and set up a window */
    glutCreateWindow("hello, teapot!");
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);

    /* define the projection transformation */
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-2.0,2.0,-2.0,2.0,-2.0,2.0);

    /* tell GLUT to wait for events */
    glutMainLoop();
}
```



7. Press the play button. You should see a lovely teapot rendered for you as per this figure on the right, and the following message in the Xcode terminal: `Checking if FLTK works: 0`

Installation on Windows 10:

1. Download Visual Studio **Community (not VSCode).** This is free and you may need to sign in so that it doesn't keep reminding you of the 1 month license.
2. If C++ is not an already installed language in Visual Studio, you need to install **Desktop development with C++** through the Visual Studio Installer
3. Now check if C++ is running correctly first.
    1. Create a new project, select template as Visual C++
    2. Select Console App, to run code in a Windows terminal, give it any name such as Project1.cpp. Select a project location that you want.
    3. Tick the box: Place solution and project in the same directory
    4. Go to the main.cpp file (or whichever .cpp file that contains the main() function) and paste the following code:

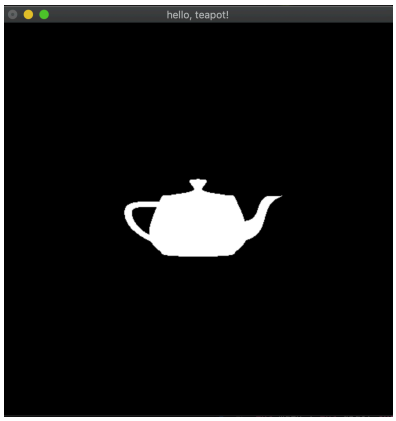
```
#include <iostream>
using namespace std;

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

    5. Click the play button (run) and you should see "Hello World" printed on the console. If this is OK, move on to the next step
    6. If you encounter any bugs, this website: https://visualstudio.microsoft.com/vs/support/hello-world-c-using-visual-studio-2017/ shows the step by step solution using VS2017 for your reference.
4. Now, download FreeGlutFLTK.zip from e-dimension and unzip it. You shall find 3 folders called FL, GL, and lib, and a file called freeglut.dll.
5. In your project folder you just created above (the folder containing the .sln),
    1. Copy the three folders: FL, GL, and lib, and freeglut.dll over to the folder where your .sln reside.
    2. Also, paste freeglut.dll inside the Debug folder. This is the folder where you should encounter the .exe (executable) of the Hello World! above, named something like Project1.exe.
    3. Press Alt+Enter to open the Properties of your VS project, and do these three things:

1. Go to VC++ Directories, under Library Directories add the path to the `lib` folder that you just copied in in your project. This tells the IDE where to find your .lib files, which you will add as under *linker*
2. Go to C/C++/General, under Additional Include Directory, include your project folder, i.e: that folder that contains your .sln. This tells the IDE where to find FL and GL folder.
3. Go to Linker, and click Input under it. Modify the Additional dependencies to include: `fltkd.lib, fltkgld.lib,` and `freeglut.lib`
4. Paste the following code to overwrite the **entire** main.cpp file (or whatever file containing the main() function above:

```
#include <windows.h>
#include <stdio.h>
#include <GL/gl.h>
#include <GL/glut.h>
#include <FL/Fl.H>

void display () {

    /* clear window */
    glClear(GL_COLOR_BUFFER_BIT);

    /* future matrix manipulations should affect the modelview matrix */
    glMatrixMode(GL_MODELVIEW);

    /* draw scene */
    glutSolidTeapot(.5);

    /* flush drawing routines to the window */
    glFlush();

}

void reshape ( int width, int height ) {

    /* define the viewport transformation */
    glViewport(0,0,width,height);

}

int main ( int argc, char * argv[] ) {

    unsigned eventCoordX = Fl::event_x();
    printf("Checking if FLTK works: %d\n", eventCoordX);

    /* initialize GLUT, using any commandline parameters passed to the
       program */
    glutInit(&argc,argv);

    /* setup the size, position, and display mode for new windows */
    glutInitWindowSize(500,500);
    glutInitWindowPosition(0,0);
    glutInitDisplayMode(GLUT_RGB);

    /* create and set up a window */
    glutCreateWindow("hello, teapot!");
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);

    /* define the projection transformation */
```

```
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-2.0,2.0,-2.0,2.0,-2.0,2.0);

    /* tell GLUT to wait for events */
    glutMainLoop();
}
```



6. You shall be greeted with a window and a lovely white pot rendered in it and the following printed message on the terminal: `Checking if FLTK works: 0`

Adding vecmath module into your project

Somewhere down the line, for the assignments, you need to also add the vecmath module so that you have the data structure to represent all types of vectors, matrices, and their operations. The .cpp files and the header files are contained in the vecmath.zip that you can download from e-dimension. You can treat it as a normal .h and .cpp files integrated into your project so that you can call them from your main file or other .cpp files. To integrate it to your project, after you follow all the steps above, do:

1. For MacOS, click File >> add Files to Assignment 1 and add **all** the header files and the .cpp files of vecmath into your project. Then you can treat it as normal files that you can include using: include “vecmath.h” with the “” and not <>. You can regroup them if you want after you import them, so that the are neatly arranged in your project folder.

2. For Windows 10 user, paste the entire vecmath folder in your project directory, where your .sln is. Then, press Alt + Enter, to open the properties windows. Under C/C++, additional include directories, add `vecmath/include.` Press OK, and then go to your project directory again, the Source Files, right click and add **all** the header files and the .cpp files of vecmath inside the vecmath/include and vecmath/src folders.