CMPE 360

Fall 2023

Project 6

Hello WebGL

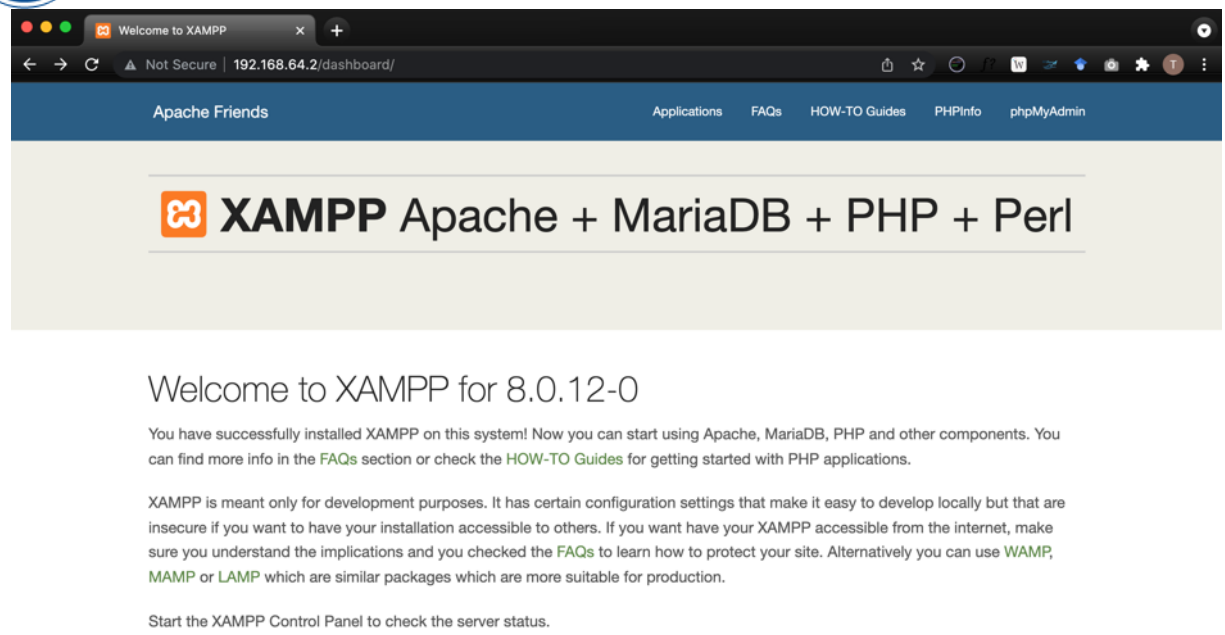
Follow the instruction carefully. Be aware of the Checkpoints below. Make sure you complete each one since we will do the grading based on them.

WebGL Basics

We will use WebGL in the course, as it is the most stable and portable version of real-time graphics libraries (among OpenGL, Metal, etc.). WebGL provides a low-level graphics API that is very similar to these other libraries, but it can run on a common browser that supports Javascript. In fact, all modern mobile phones, tablets and computers come with WebGL pre-installed.

Step 1: Install a web server

Although many WebGL functionalities work without a web server, some tasks (e.g. reading a texture file) requires a web server running locally on your computer. If you have already worked with web servers, or use a Python environment, you may already have a web server installed on your computer. But there are many easy-to-use web servers that you may use. One example is XAMPP.



☐ Checkpoint 1: Install a web server and include a snapshot of the local web server output (here is the example from XAMPP).

Step 2: Read the JavaScript / WebGL Tutorial

The website http://learnwebgl.brown37.net/index.html contains an introductory tutorial on JavaScript and Web programming (including HTML, DOM, events).

You may already be familiar with JavaScript from your previous works. In the next assignments we will use JavaScript for programming WebGL applications, but we will use a minimal subset of JavaScript.

In this step, read the Section 1 (“The Big Picture”), Section 2 (“Tools and Languages”), and Section 3 (“Model Data”). Section 3 is basically a review of what we discussed in the context of ray tracing, but it contains many useful information about how WebGL represent them.

Step 3: Write a simple WebGL application that contains

In this step, our goal is to get comfortable drawing with vertex buffers, uniforms and shaders by:

- drawing several things with separate draw commands
- using different primitives
- changing default line and point sizes
- changing colors on the fly

For this step, we will use the lab sheet at the following web page as the starting point:

- https://www.labs.cs.uregina.ca/315/WebGL2/Lab2/
- https://www.labs.cs.uregina.ca/315/WebGL2/Lab1/#HTML_TEMPLATE
- https://www.labs.cs.uregina.ca/315/WebGL2/Lab1/#JS_TEMPLATE

The requirements for Step 3 are:

i. **Draw a picture** that contains at least three of the various OpenGL primitives. It should look very different from any in-lab demonstrations.

   - must use POINTS
   - must use at least one line type: either LINES, LINE_STRIP, or LINE_LOOP (not including the line loop in the instructions above)
   - must use at least one polygon type: either TRIANGLES, TRIANGLE_STRIP, or TRIANGLE_FAN (not including any geometry used as an in-lab demo)

ii. **Use at least 3 different colours**. Do this with a uniform shader variable or a vertex colour array input.

iii. **Use at least two point sizes**. A uniform must be set up to allow you to set the point size from JavaScript.

iv. **Write code that would draw at least two different line sizes**. Tell us in a JavaScript comment whether or not it worked, and on what OS+graphics card combination.

v. **Artistic impression** - your drawing should resemble something and reflect some effort.

Hint: the circle function shown in the lab notes can easily be modified to create points for arbitrarily sized and centered circles. Using different sides arguments you can draw triangles, squares, pentagons, hexagons, etc. You can learn the size of the array you get back with its .length member. You can create a same size color array with a loop. You can concatenate it with other arrays to add it to your array buffers. Its points should work well with both LINE_LOOP and TRIANGLE_FAN. TRIANGLES might produce a neat effect too.

Samples of previous work is at the link below. We will make a gallery of best images at the end of the course:

https://www.labs.cs.uregina.ca/315/WebGL2/Lab2/gallery.html

☐ Checkpoint 3: Save the image in the PNG or JPG format.

PROJECT REPORT SUBMISSION

!!IMPORTANT!!

This project report is due by 23:59 on Sunday, 3 December 2023.

Follow the instruction and prepare a project report pdf file for uploading to the LMS, your pdf file should include all parts. Please make sure your answers are numbered as below:

PART 1

- Save the image of the running local web server on your pdf file.

PART 2

- Save your image on your pdf file. Explain how your image satisfies homework requirements in itemized form. **Explain your process in detail.**
- Also include the image as a separate file in PNG or JPG format as part of your submission.
- Also include you complete **JavaScript + HTML files** that can run as is on our computers. Also include other folders (e.g. Common folder) you have used,

After finishing the project, please upload a single ZIP file, which includes a separate PNG/JPG file of your Step 3 results, and your **complete JavaScript + HTML files,** to the course's LMS site. You will upload your project report to "Project6 Report Submission"s part on LMS.

Grading Rubric

PART 1	**Points**
Save the image of the running local web server on your pdf file	10 Points
PART 2	
Save your image on your pdf file. Explain how your image satisfies homework requirements in itemized form. Explain your process in **detail**.	40 points
Also include the image as a separate file in PNG or JPG format as part of your submission.	10 points
Also include you complete **JavaScript + HTML files** that can run as is on our computers.	10 points
Quiz	30 points