# Support for Geometry and Tessellation Shaders In The Processing Platform

---


### Name: James Muir

### Email: zyborgjames@gmail.com

### Processing Forum: zyborg


---

## Project Abstract


Enhance the Processing platform shader implementation by adding support for geometry and tessellation shaders to the P2D and P3D OpenGL renderers.


---

## Project Description

My project will enable the use of geometry and tessellation shaders when using the P2D and P3D OpenGL renderer modes in Processing. I will create a library to wrap some existing functionality, code new functionality, and expose the necessary OpenGL APIs to enable simpler use of advanced shader techniques. I will test the new functionality and existing shader support by writing several reference samples and provide end user documentation for this new functionality.

Adding support for geometry and tessellation shaders will give greater functionality to the current default rendering pipeline. Geometry shaders will allow users to generate scene geometry on the graphics card itself leveraging current hardware capabilities. Tessellation shaders will allow better Level of Detail (LoD) to be used with terrain and objects in order to improve performance.

Basically, I plan to add to the programmable pipeline rendering capabilities and provide advanced shader tutorials and examples for the Processing platform.

---

## Development Process

Timeline:

- Before May 30:
    - Familiarize myself with the Processing OpenGL codebase

    - Research best methods of implementing geometry and tessellation shaders
    - Examine examples by codeanticicode
- May 30 - June 26:
    - Use a test driven development approach to create a library that adds support for geometry and tessellation shaders
    - Do research and experiments to understand current code implementation
    - Define any needed new wrappers and APIs
    - Develop and test samples to prove functionality
    - Document
- June 26 - July 28:
    - Use a test-driven development approach to create default shaders for both geometry and tessellation
    - Research examples of tessellation and geometry shaders (such as these by codeanticode)
    - Develop and test a reference implementation to prove functionality
    - Document
- July 28 - August 29:
    - Polish shaders
    - Test
    - Fix bugs
    - Manage peer review of code samples and documentation
    - Complete end user documentation (similar to the current shader tutorial)
    - Write more sample shaders, if possible
    - Finalize and commit all code
- Stretch Goals:
    - If things go much faster than expected, I can also potentially work on adding support for multiple rendering targets.

---

## More about me

I will graduate this June from Bainbridge Island High School in Washington State. I am currently a 4.0 student and National Merit finalist. My favorite subjects are Math, Physics, and Engineering, and I have taken AP calculus, statistics, physics and chemistry and last summer I

attended the Carnegie Mellon AP/EA program in Pittsburgh where I completed CS 15-112 (a computer science course) for college credit.  My final project there was a 3D rendering engine for procedural terrain generation created with Python leveraging OpenGL.

I will be a freshman at the University of Washington this fall. I was accepted directly into the Mechanical Engineering program and am interested in Mechatronics.

I currently enjoy programming as a hobby. I am particularly fascinated by 3D graphics; all of the geometry and cool algorithms are intriguing. Most of my experience is in Python, with a bit of OpenGL and GLSL knowledge, as well as some C#.

I think that the Processing project is a great way to dig deeper into coding, especially for people who like visual feedback, like me. I have not personally used Processing much yet, but since I discovered it I am sure that I will use it much more in the future. I love how quickly I can get something on the screen with Processing. I have also used a derivative of the Processing IDE to program an Arduino microcontroller.

I am an active beta tester and I've submitted many bug reports for software that I use. I am familiar with test driven development and with the process of isolating issues when debugging, as well as robust software documentation procedures. Of course, debugging can be more difficult when the output is visual, but that can also be used to your advantage, by rendering debug information to the screen itself.

My github account currently contains 2 repositories. One is my final project for my CS course at Carnegie Mellon, where I created a 3D graphics engine from scratch, and used OpenGL to render a basic scene that I procedurally generated using simplex noise. The 3D engine that I created used quaternions to handle rotation, for which I wrote my own library. The other is a very basic start to a project that aims to model the effects of genetics and selective pressures on a population over time.

I am very motivated to work on a substantial and interesting project this summer to both enhance my coding skills, solve some interesting problems, work with interesting people, and contribute my own code to the open source community. I am also interested in doing some bug fixing and sample code development in other parts of the processing platform if I am needed, I am flexible.