Programming The Pipeline

- Review the Pipeline

Vertex Stage: < input: world space vertices
                  output: NDC vertices w/ more info

Assembled/Primitives Stage: input: triangles clip/subdivide
                             output: triangles

Rasterization - find fragments
              interpolate

Fragment < input: fragments
          output: fragments "filled in" (~

Vertex in Fragment

Input: partially filled object
Output: fill in key slots

Language Detail (instantiates model)

model gives variable types
uniform (one per primitive)
vertex (or input) < call in

Varying variables < called "out" from Vertex shaders
How GLSL does it
- everything passed through variables

Shaders are all:
void main () {
    all they do is fill in the magic variables

Some variables are defined by GL
Some variables can be defined by user

Variables always "connect" 2 things

C++ → Vertex → Fragment
| attributes | varying |
| uniforms   |         |

built in:
- MVMatrix, MVPMatrix, Normal Matrix
- uniforms
- gl_Vertex, gl_Color, gl_Normal, gl_MultiTexCoord
- attribute
- gl_Position, gl_FrontColor
- outputs
- gl_FragColor, gl_FragDepth

see Appendix I of red book
Doing this:

1. Load in source code to shader
2. Run compiler (check for errors)
3. Attach shaders
4. Draw polygon \(\Rightarrow\) Send variables
5. Detach shader

Mechanism to see what variables shader wants:
- Name \(\Rightarrow\) Index (per program)
- Array of attributes/uniforms