WebGL Fragment Shader Profiler

Terry Sun & Sally Kong, CIS565 Final Project

Project Idea

- A tool for profiling fragment shaders!
- A Chrome extension (!) which interacts with the shaders on a page and can profile them (semi-automatically) to show you which sections are taking longer.
- Mouse over the page to see hotspots in particular pixel.

This Milestone

- Research/proof of concept better shader modification
  - With AST analysis
- Research/proof of concept for mouse input & pixel selection
- Experiment with ShaderEditor Extension

Progress

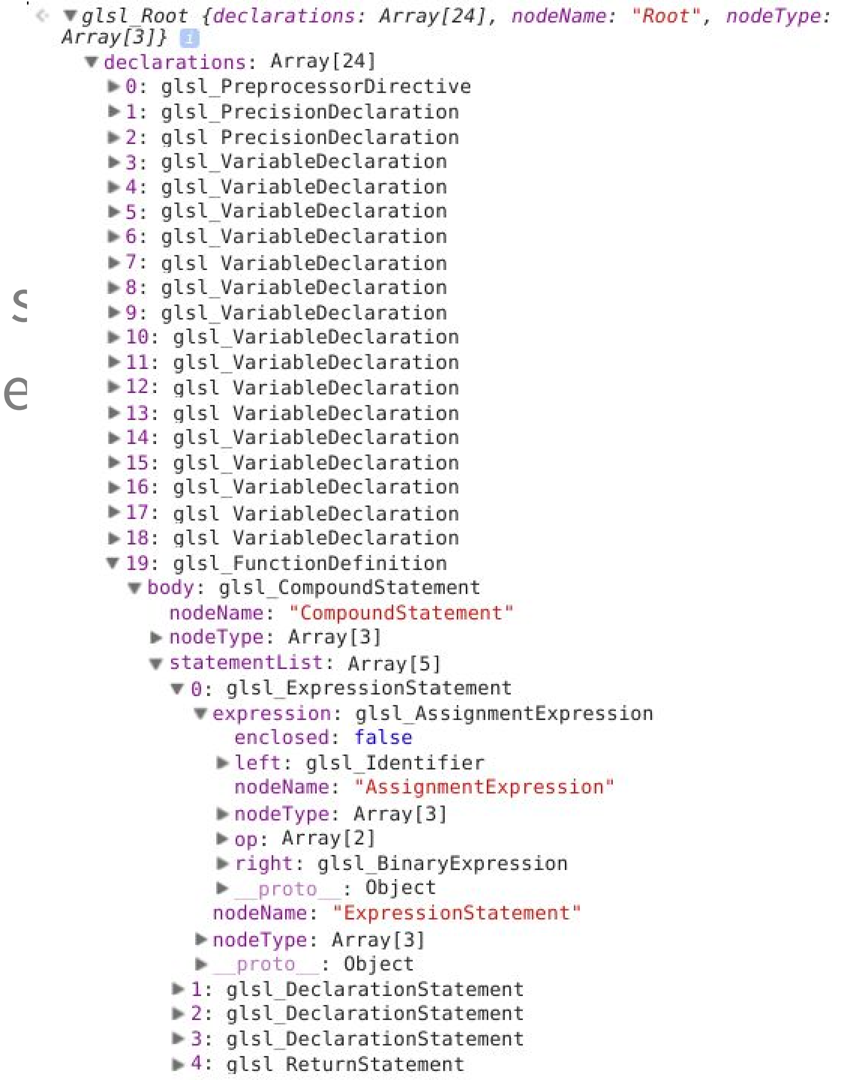
- + Gained new party member!

Progress

- Research/proof of concept better shader modification
  - There are like 12 JS GLSL parsers which don’t work for various reasons
  - haxe-glsl-parser

Progress

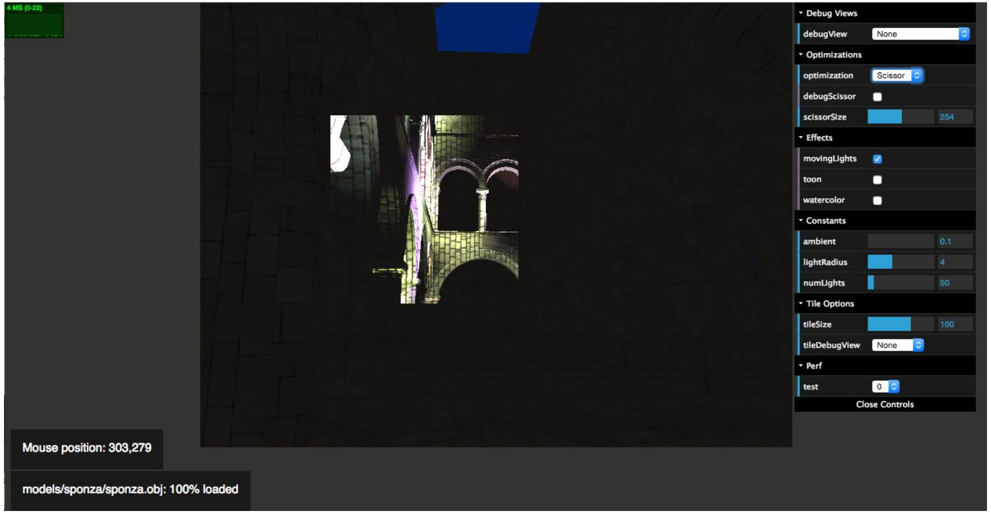
- Research/proof of concept better s
  - There are like 12 JS GLSL parse various reasons
  - haxe-glsl-parser



```
▼glsl_Root {declarations: Array[24], nodeName: "Root", nodeType:
Array[3]}
  ▼declarations: Array[24]
    ▶0: glsl_PreprocessorDirective
    ▶1: glsl_PrecisionDeclaration
    ▶2: glsl_PrecisionDeclaration
    ▶3: glsl_VariableDeclaration
    ▶4: glsl_VariableDeclaration
    ▶5: glsl_VariableDeclaration
    ▶6: glsl_VariableDeclaration
    ▶7: glsl_VariableDeclaration
    ▶8: glsl_VariableDeclaration
    ▶9: glsl_VariableDeclaration
    ▶10: glsl_VariableDeclaration
    ▶11: glsl_VariableDeclaration
    ▶12: glsl_VariableDeclaration
    ▶13: glsl_VariableDeclaration
    ▶14: glsl_VariableDeclaration
    ▶15: glsl_VariableDeclaration
    ▶16: glsl_VariableDeclaration
    ▶17: glsl_VariableDeclaration
    ▶18: glsl_VariableDeclaration
    ▼19: glsl_FunctionDefinition
      ▼body: glsl_CompoundStatement
          nodeName: "CompoundStatement"
        ▶nodeType: Array[3]
        ▼statementList: Array[5]
          ▼0: glsl_ExpressionStatement
            ▼expression: glsl_AssignmentExpression
                enclosed: false
              ▶left: glsl_Identifier
                nodeName: "AssignmentExpression"
              ▶nodeType: Array[3]
              ▶op: Array[2]
              ▶right: glsl_BinaryExpression
              ▶__proto__: Object
              nodeName: "ExpressionStatement"
            ▶nodeType: Array[3]
            ▶__proto__: Object
          ▶1: glsl_DeclarationStatement
          ▶2: glsl_DeclarationStatement
          ▶3: glsl_DeclarationStatement
          ▶4: glsl_ReturnStatement
```

Progress

- Research/proof of concept for mouse input & pixel selection.
  - Demo!



debugView: None

optimization: Scissor

debugScissor: □

scissorSize: 254

movingLights: ☑

toon: □

watercolor: □

ambient: 0.1

lightRadius: 4

numLights: 50

tileSize: 100

tileDebugView: None

test: 0

Mouse position: 303,279

models/sponza/sponza.obj: 100% loaded

Next Milestone

- AST-based shader modification.
  - User markup with `#pragma`.
  - Define heuristics for automatic analysis.
    - I have no idea if this will work at all.
- Profiling with GL Injection.
  - Doesn’t require user to add library calls to their JS.
  - Hijacks and wraps gl calls (thanks kai)