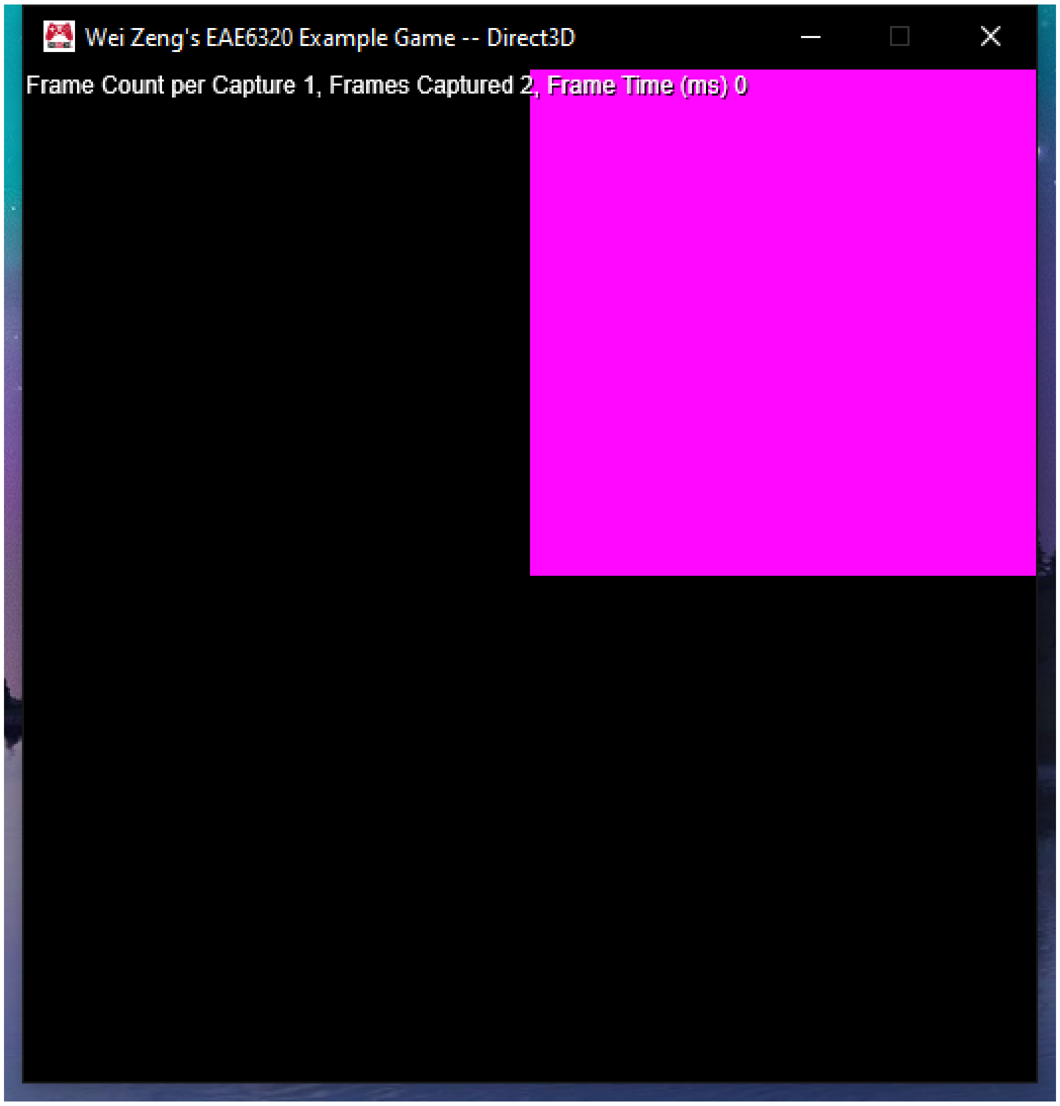
Assignment 2 Writeup

Game Executable:

https://github.com/WayGold/EAE6320_Assignments/blob/main/MyGame_.zip

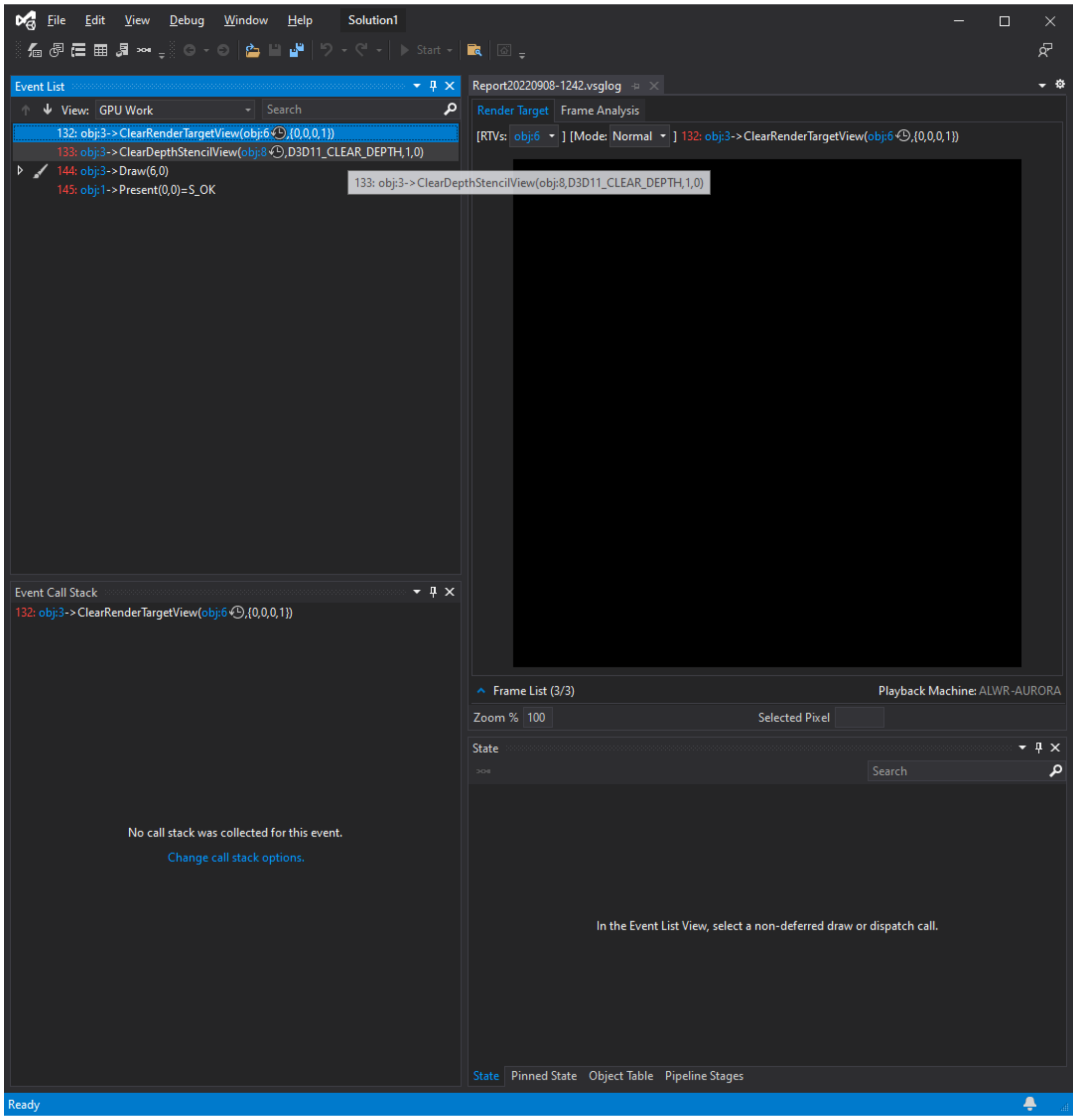
Game running:



Frame Count per Capture 1, Frames Captured 2, Frame Time (ms) 0

(Figure 1: two triangles animation)

Direct3D GPU capture:



Event List

View: GPU Work

132: obj:3->ClearRenderTargetView(obj:6,{0,0,0,1})
133: obj:3->ClearDepthStencilView(obj:8,D3D11_CLEAR_DEPTH,1,0)
144: obj:3->Draw(6,0)
145: obj:1->Present(0,0)=S_OK

133: obj:3->ClearDepthStencilView(obj:8,D3D11_CLEAR_DEPTH,1,0)

Report20220908-1242.vsglog

[RTVs: obj:6] [Mode: Normal] 132: obj:3->ClearRenderTargetView(obj:6,{0,0,0,1})

Frame List (3/3) Playback Machine: ALWR-AURORA

Zoom % 100 Selected Pixel

Event Call Stack

132: obj:3->ClearRenderTargetView(obj:6,{0,0,0,1})

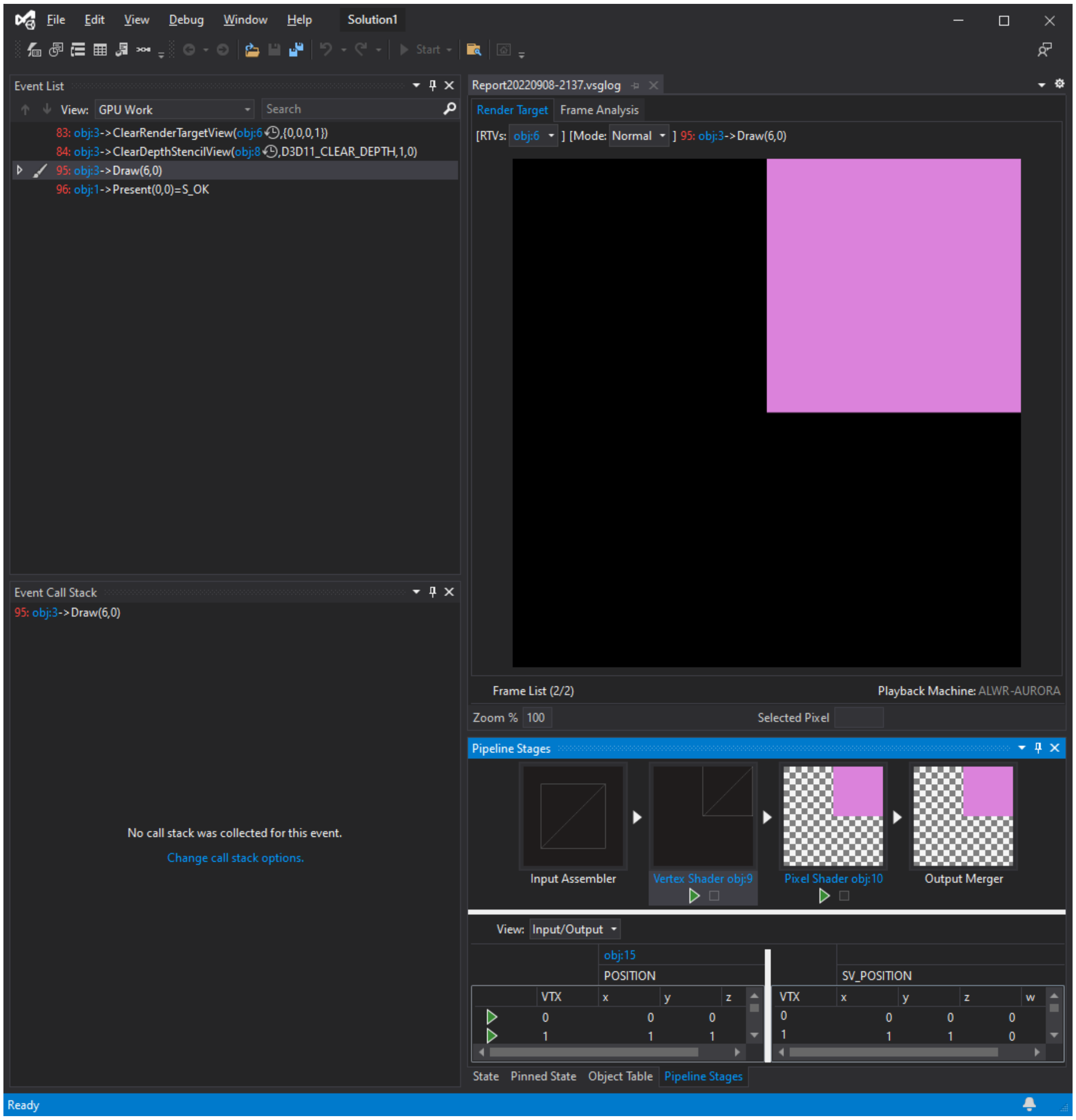
No call stack was collected for this event.

Change call stack options.

State

In the Event List View, select a non-deferred draw or dispatch call.

(First Capture: render target when it is all black)



83: obj:3->ClearRenderTargetView(obj:6,{0,0,0,1})
84: obj:3->ClearDepthStencilView(obj:8,D3D11_CLEAR_DEPTH,1,0)
95: obj:3->Draw(6,0)
96: obj:1->Present(0,0)=S_OK

Event Call Stack
95: obj:3->Draw(6,0)

No call stack was collected for this event.
Change call stack options.

Report20220908-2137.vsglog
[RTVs: obj:6] [Mode: Normal] 95: obj:3->Draw(6,0)
Frame List (2/2) Playback Machine: ALWR-AURORA
Zoom % 100 Selected Pixel

Pipeline Stages: Input Assembler, Vertex Shader obj:9, Pixel Shader obj:10, Output Merger

View: Input/Output

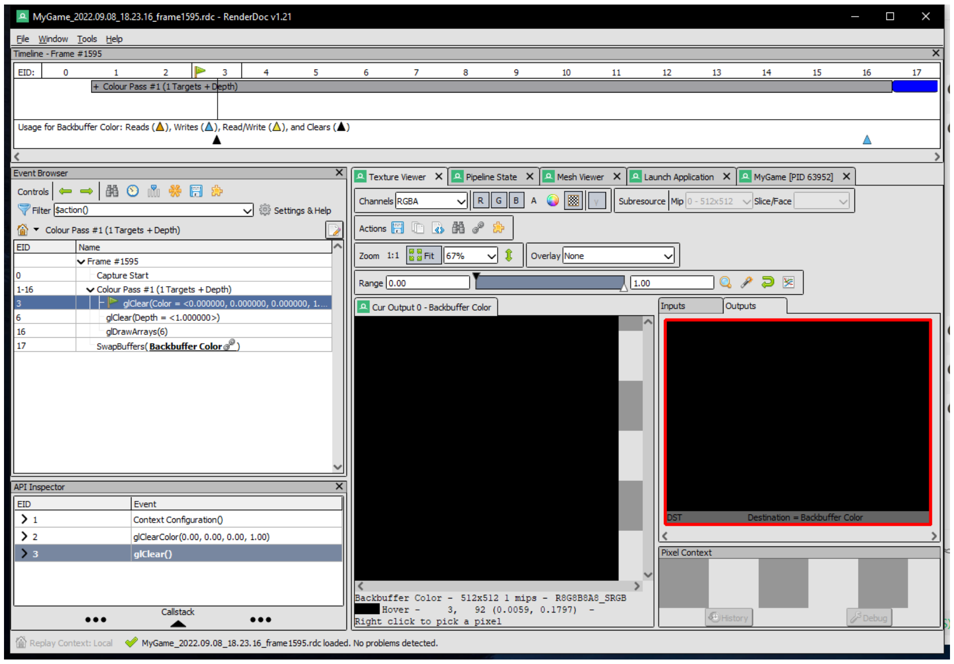
obj:15

POSITION VTX	x	y	z
0	0	0	
1	1	1	

SV_POSITION VTX	x	y	z	w
0	0	0	0	
1	1	1	0	

(Second Capture: render target with the mesh)

OpenGL GPU capture in RenderDoc:



MyGame_2022.09.08_18.23.16_frame1595.rdc - RenderDoc v1.21

Usage for Backbuffer Color: Reads (△), Writes (△), Read/Write (△), and Clears (▲)

Filter $action()

Colour Pass #1 (1 Targets + Depth)

EID	Name
	Frame #1595
0	Capture Start
1-16	Colour Pass #1 (1 Targets + Depth)
3	glClear(Color = <0.000000, 0.000000, 0.000000, 1.…
6	glClear(Depth = <1.000000>)
16	glDrawArrays(6)
17	SwapBuffers(**Backbuffer Color**)

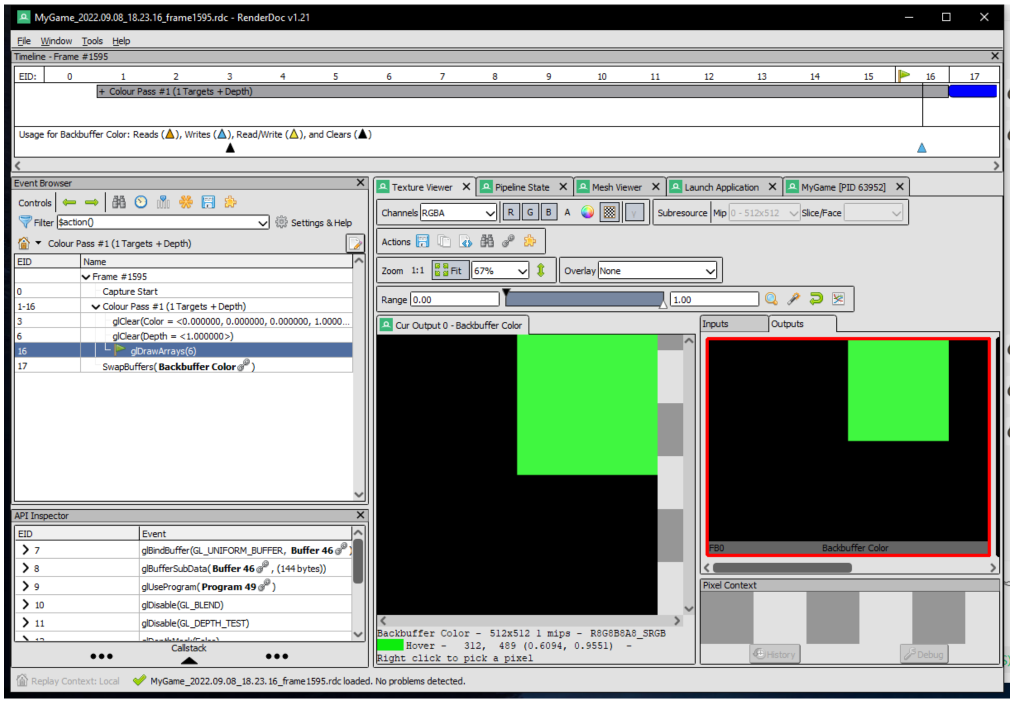
API Inspector

EID	Event
1	Context Configuration()
2	glClearColor(0.00, 0.00, 0.00, 1.00)
3	glClear()

```
Backbuffer Color - 512x512 1 mips - R8G8B8A8_SRGB
   Hover -      3,   92 (0.0059, 0.1797)  -
Right click to pick a pixel
```

MyGame_2022.09.08_18.23.16_frame1595.rdc loaded. No problems detected.

(First Capture: render target when it is all black)



MyGame_2022.09.08_18.23.16_frame1595.rdc - RenderDoc v1.21

Usage for Backbuffer Color: Reads (△), Writes (△), Read/Write (△), and Clears (▲)

Filter $action()

Colour Pass #1 (1 Targets + Depth)

EID	Name
	Frame #1595
0	Capture Start
1-16	Colour Pass #1 (1 Targets + Depth)
3	glClear(Color = <0.000000, 0.000000, 0.000000, 1.0000…
6	glClear(Depth = <1.000000>)
16	glDrawArrays(6)
17	SwapBuffers(**Backbuffer Color**)

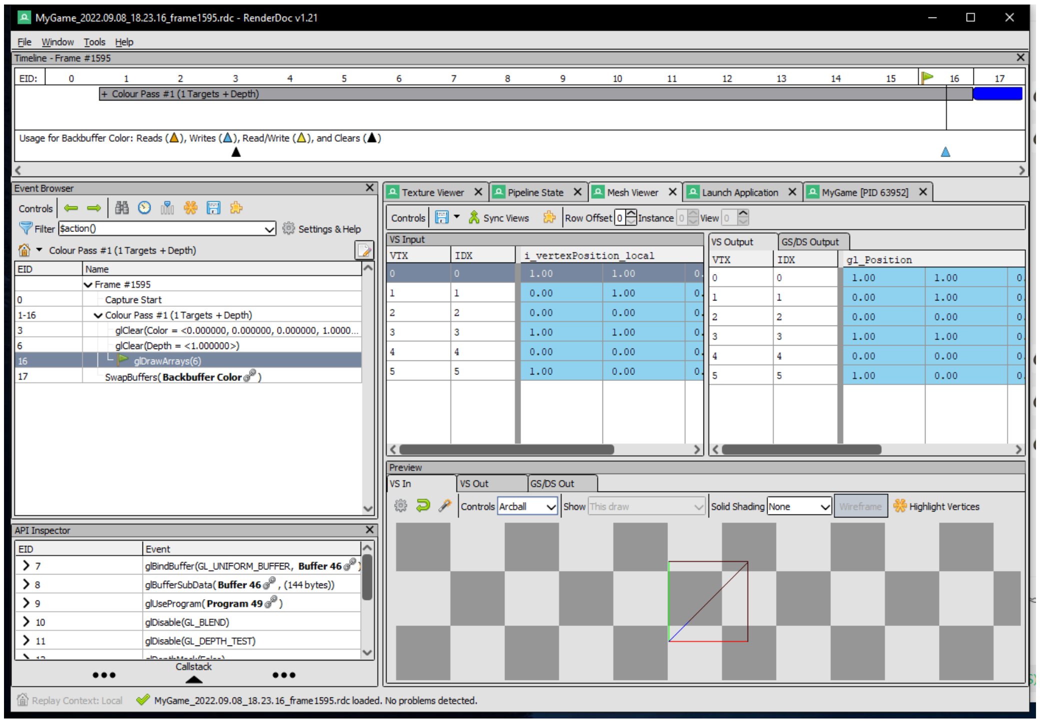
API Inspector

EID	Event
7	glBindBuffer(GL_UNIFORM_BUFFER, **Buffer 46**)
8	glBufferSubData(**Buffer 46**, (144 bytes))
9	glUseProgram(**Program 49**)
10	glDisable(GL_BLEND)
11	glDisable(GL_DEPTH_TEST)

```
Backbuffer Color - 512x512 1 mips - R8G8B8A8_SRGB
   Hover -    312,  489 (0.6094, 0.9551)  -
Right click to pick a pixel
```

MyGame_2022.09.08_18.23.16_frame1595.rdc loaded. No problems detected.

(Second Capture: render target with the mesh)



Usage for Backbuffer Color: Reads (▲), Writes (▲), Read/Write (▲), and Clears (▲)

Event Browser

Filter $action()

Colour Pass #1 (1 Targets + Depth)

EID	Name
	Frame #1595
0	Capture Start
1-16	Colour Pass #1 (1 Targets + Depth)
3	glClear(Color = <0.000000, 0.000000, 0.000000, 1.0000...
6	glClear(Depth = <1.000000>)
16	glDrawArrays(6)
17	SwapBuffers(**Backbuffer Color**)

VS Input

VTX	IDX	i_vertexPosition_local		
0	0	1.00	1.00	0.
1	1	0.00	1.00	0.
2	2	0.00	0.00	0.
3	3	1.00	1.00	0.
4	4	0.00	0.00	0.
5	5	1.00	0.00	0.

VS Output

VTX	IDX	gl_Position		
0	0	1.00	1.00	0
1	1	0.00	1.00	0
2	2	0.00	0.00	0
3	3	1.00	1.00	0
4	4	0.00	0.00	0
5	5	1.00	0.00	0

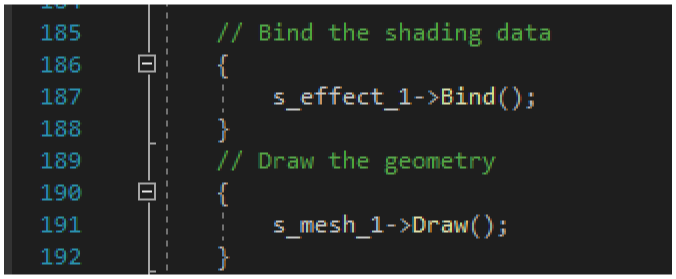
API Inspector

EID	Event
7	glBindBuffer(GL_UNIFORM_BUFFER, **Buffer 46**)
8	glBufferSubData(**Buffer 46**, (144 bytes))
9	glUseProgram(**Program 49**)
10	glDisable(GL_BLEND)
11	glDisable(GL_DEPTH_TEST)

Replay Context: Local   MyGame_2022.09.08_18.23.16_frame1595.rdc loaded. No problems detected.

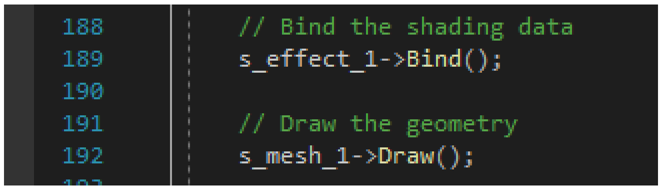
(Third Capture: show the mesh's two triangles)

Code that binds the effect and draws the mesh in Graphics.d3d.cpp/Graphics.gl.cpp:



```
// Bind the shading data
{
    s_effect_1->Bind();
}
// Draw the geometry
{
    s_mesh_1->Draw();
}
```

(Graphics.d3d.cpp)



```
// Bind the shading data
s_effect_1->Bind();

// Draw the geometry
s_mesh_1->Draw();
```

(Graphics.gl.cpp)

With the addition of the mesh and effect class, the difference between the two platform dependent Graphics.cpp files is greatly reduced. However, there are still places in each file that depend on platform specific instructions. For example, both files have platform specific instructions for clearing the color and depth buffer. The d3d graphics file also has a InitialilzeViews function with lots of d3d instructions. Like how we manage geometry and shaders, in order to eliminate the platform specific codes, we could have new interfaces for clearing color and depth buffers and refactor the platform specific codes into the newly created platform specific files, similarly for initializing view.