

Shader基础了解

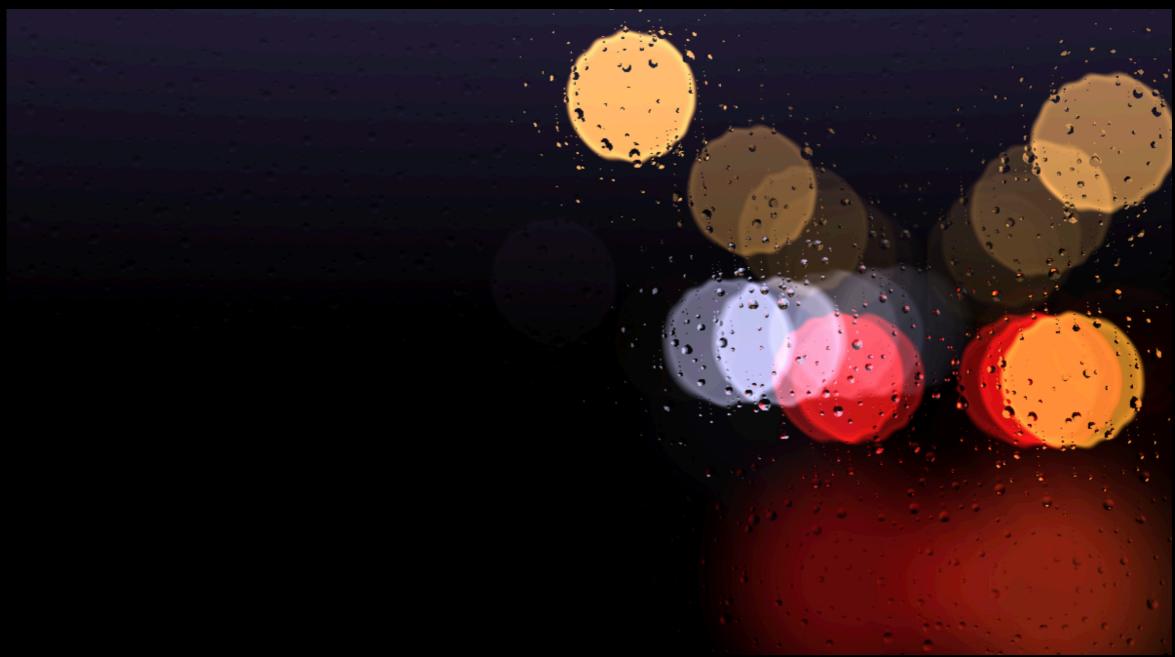
2019.4.16

- Shader是什么？
- WEBGL与Shader存在什么关系？
- 怎么将效果渲染到屏幕上？
- shader的开发语言种类
- 如何利用glsl实现一个效果

shader是什么？

shader中文名着色器，是一种计算机程序，用于图像处理（光照、亮度、颜色）等。

也可以进行cg特效处理，影片后期处理...



案例展示

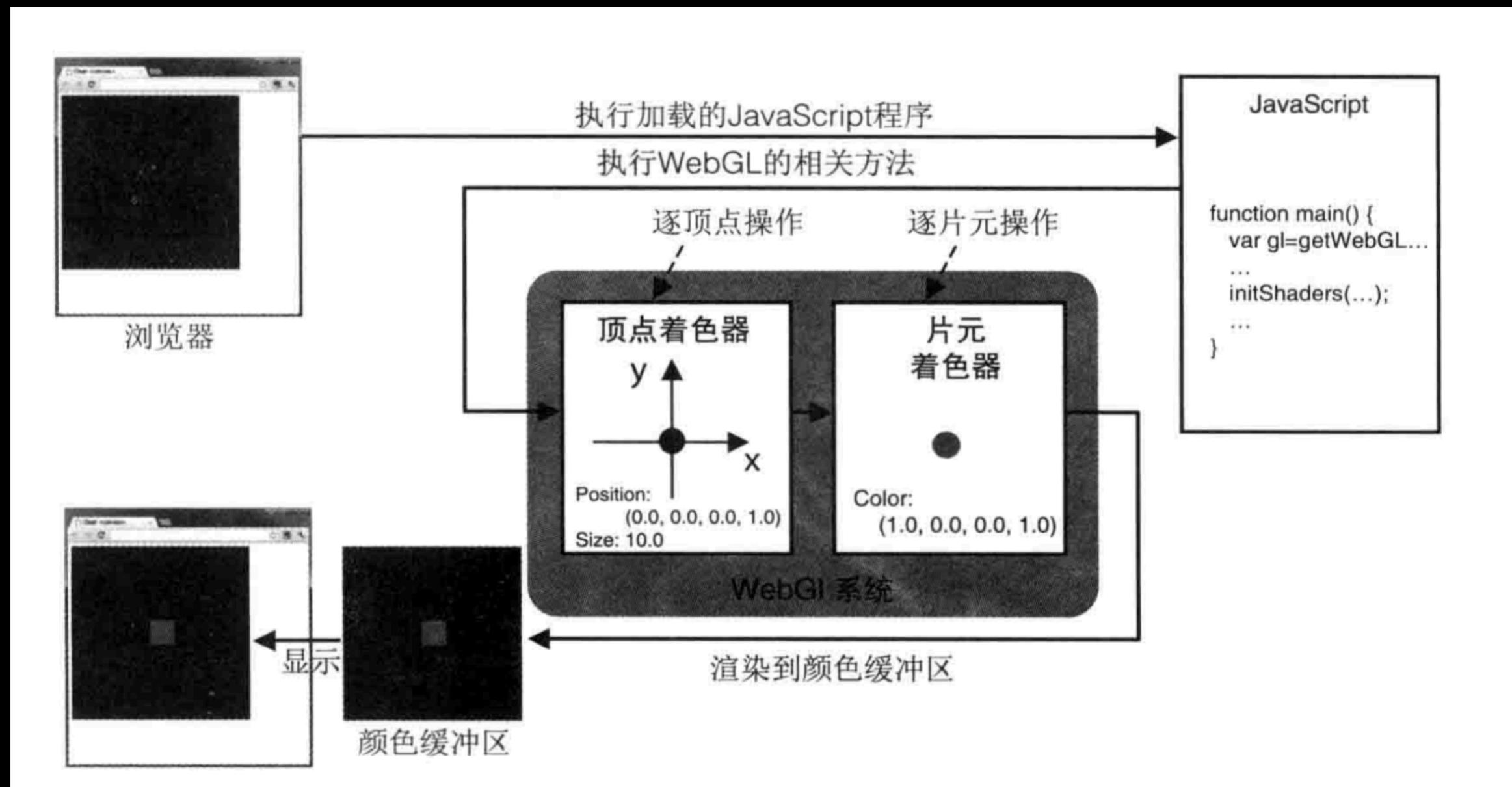
WEBGL与shader存在的关系？

WEBGL（web图形库）是一种javascript的API，用于在任何兼容的Web浏览器中呈现交互式3D和2D图形，而无需使用插件。

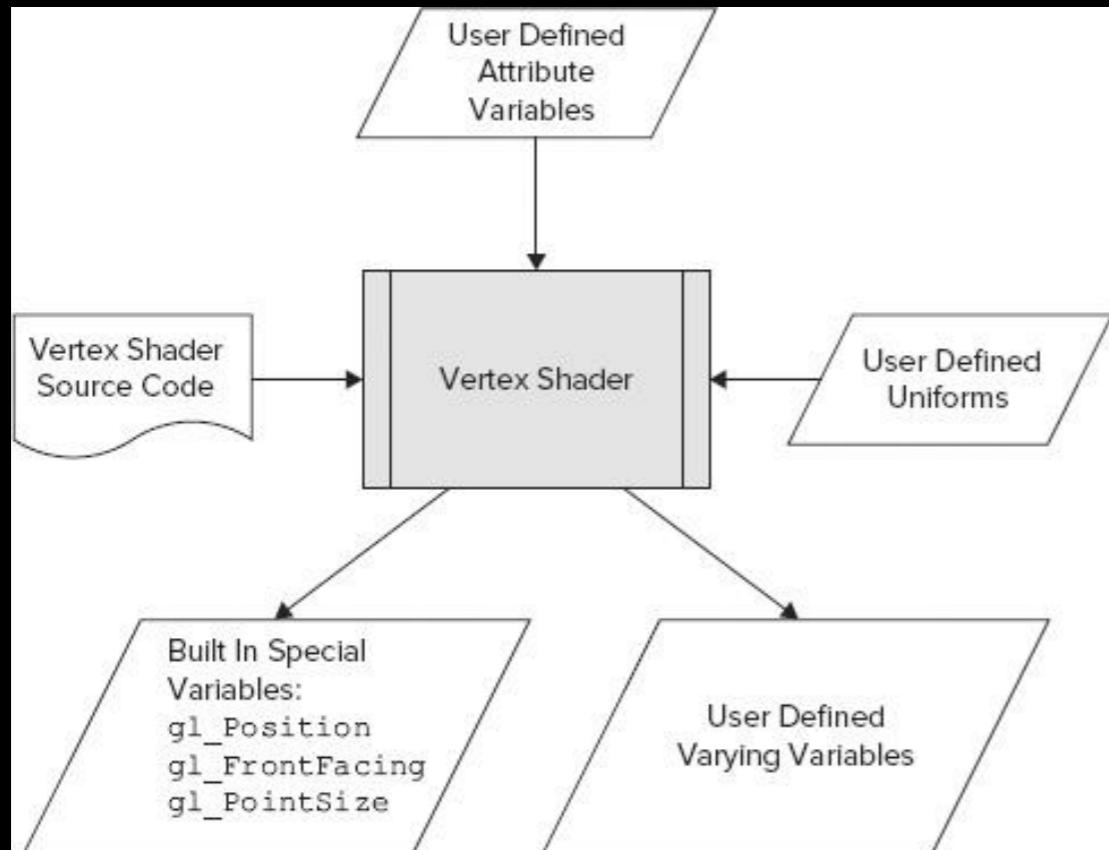
WebGL是一个光栅化引擎，它可以根据你的代码绘制出点、线、三角形（基础图形）。其中的代码输入就是shader程序



怎么将效果渲染到屏幕？

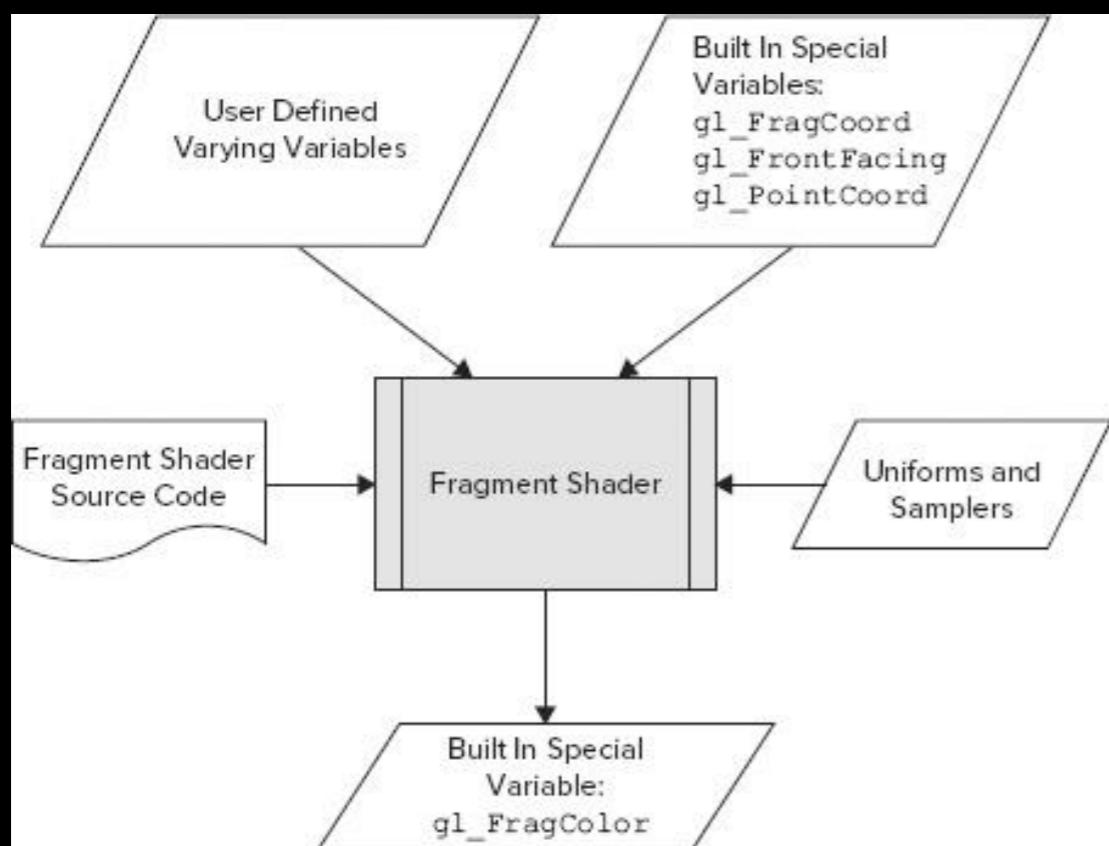


简化关系图



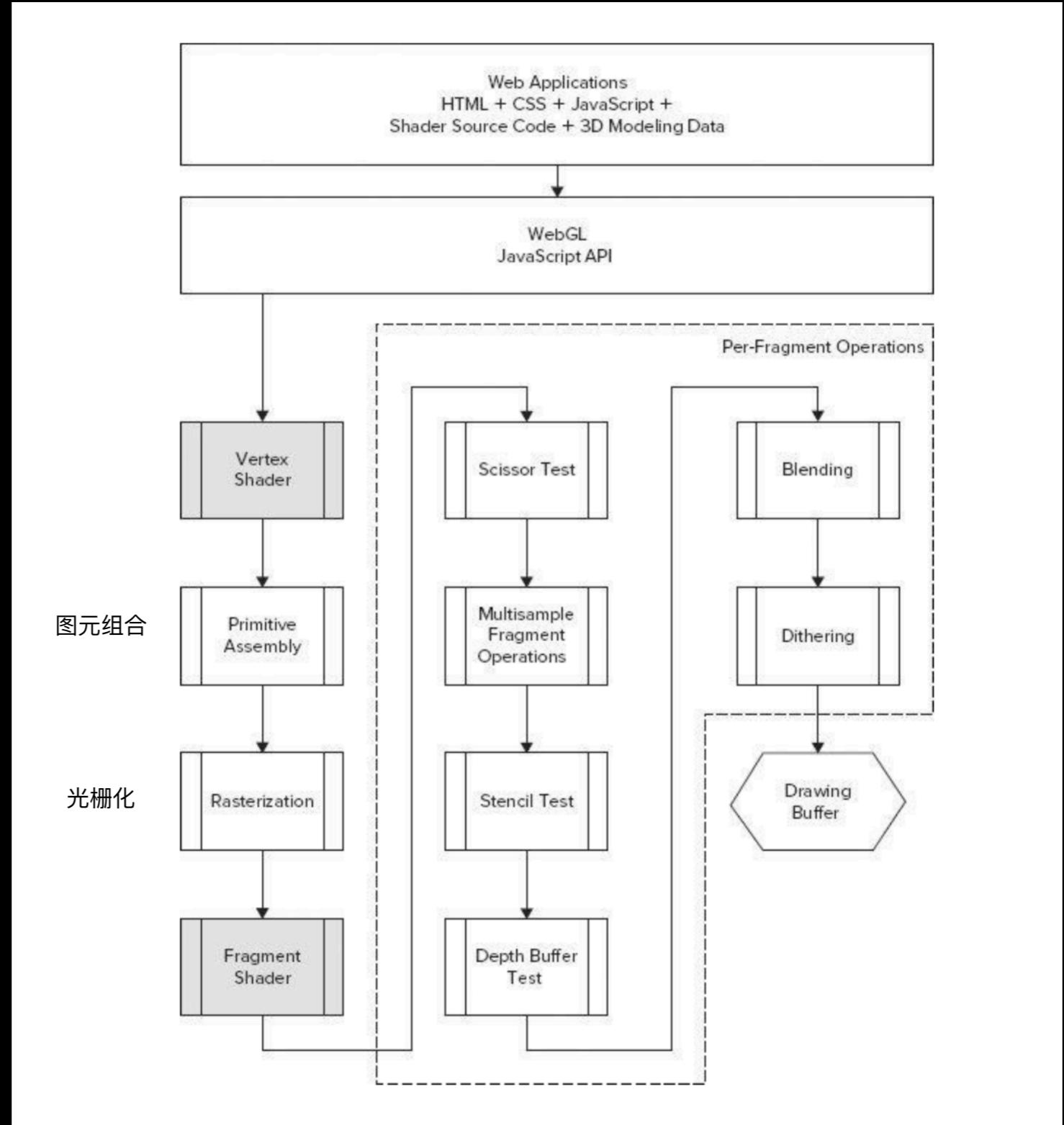
顶点着色器 (vertex shader)

- 1、顶点着色器源码
- 2、uniform 输入值
- 3、attribute 发送到GPU的一些二进制数据序列
如位置，法向量，纹理坐标，顶点颜色值等
- 4、内置变量 gl_XXX
- 5、varying 用户自定义变量



片段着色器 (fragment shader)

- 1、片段着色器源码
- 2、varying 用户自定义变量通过顶点着色器得到
- 3、uniform 输入值&纹理
- 4、内置变量 gl_XXX



详细关系图

shader开发语言选择

主要的有以下两种：

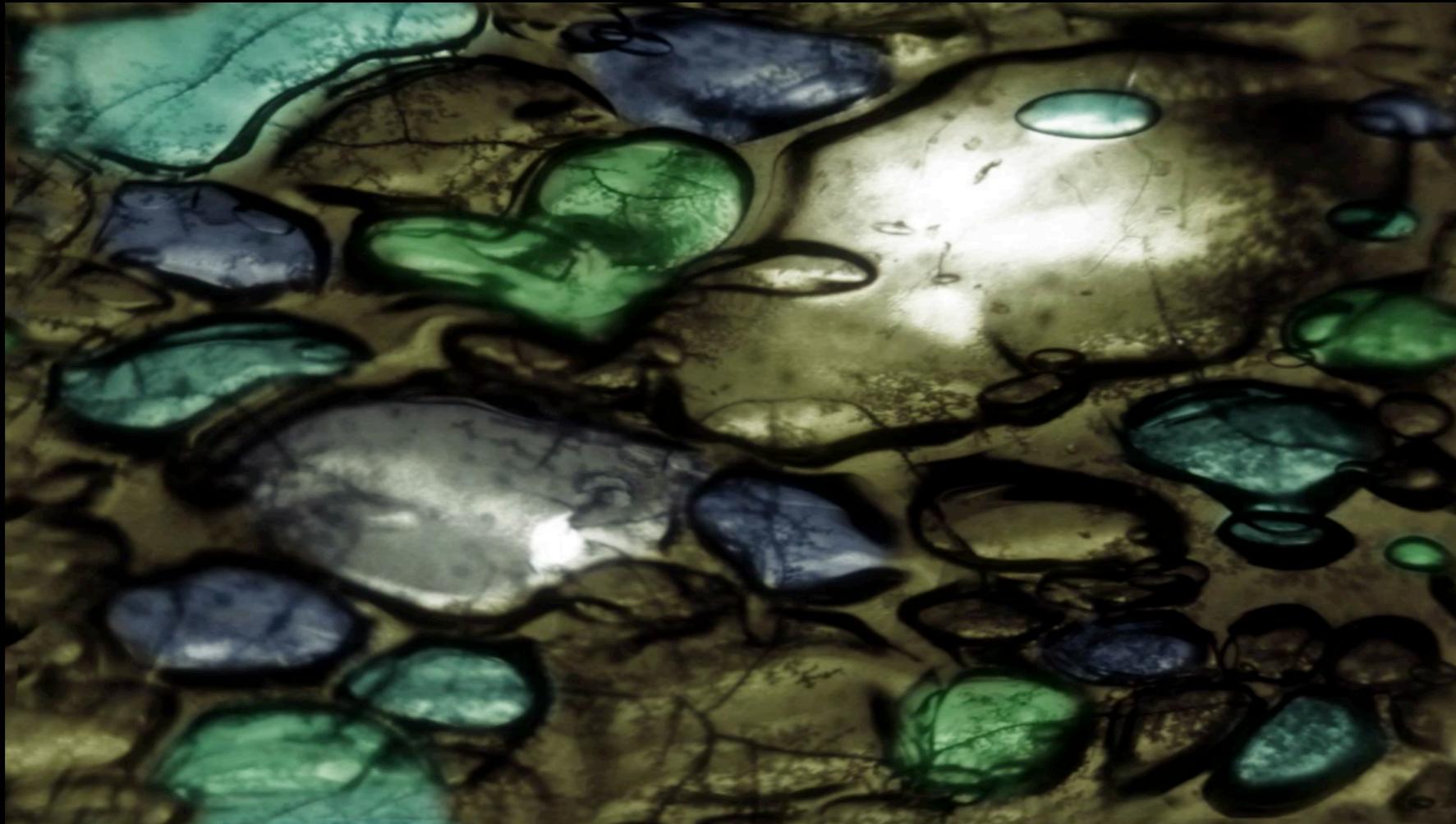
- 1、OpenGL的GLSL
- 2、DirectX的HLSL

```
varying float xxx;  
  
float easeOutBack (float t) {  
    // ..  
}  
  
float clampRange (float minValue, float maxValue, float value) {  
    // ..  
}  
  
void main () {  
    // ..  
}
```

GLSL全称是 Graphics Library Shader Language （图形库着色器语言）：

强类型语言（定义输入输出变量类型，float、int...），类似于C语言的语法，有一些内建数据类型vecN、matN

如何利用glsl实现一个效果



练习工具

shadertoy: <https://www.shadertoy.com/new>

Vscode: <https://github.com/actarian/vscode-glsl-canvas>

数学工具: <https://www.desmos.com/calculator/7j0stojccd>

图片形变简单计算公式：

$$uv.y += \sin(uv.x * a + p) * b$$

a: 频率

b: 振幅

p: 位移

怎么使用？

1、利用webgl的API构建shader运行的环境

```
<script id="vs" type="x-shader/x-vertex">
attribute vec4 a_Position;
uniform vec4 u_Translation;

void main () {
    gl_Position = a_Position + u_Translation;
}

</script>

<script id="fs" type="x-shader/x-fragment">
void main () {
    gl_FragColor = vec4(1.0, 0.0, 0.0, 1.0);
}

</script>
```

```
function getShader (id, type) {

    let shaderContent = document.getElementById(id).textContent;
    let shader = gl.createShader(gl[type]);

    // Set the shader program
    gl.shaderSource(shader, shaderContent);

    // Compile the shader
    gl.compileShader(shader);

    // Check the result of compilation
    if(!gl.getShaderParameter(shader, gl.COMPILE_STATUS)) {

        let error = gl.getShaderInfoLog(shader);

        console.log('Failed to compile shader: ' + error);

        gl.deleteShader(shader);

        return null;
    }

    return shader;
}
```

```
function initShader () {

    let fragmentShader = getShader('fs', 'FRAGMENT_SHADER');
    let vertexShader = getShader('vs', 'VERTEX_SHADER');

    //console.log(fragmentShader);
    //console.log(vertexShader);
    // Create a program object
    let program = gl.createProgram();

    // Attach the shader objects
    gl.attachShader(program, fragmentShader);
    gl.attachShader(program, vertexShader);

    // Link the program object
    gl.linkProgram(program);

    if(!gl.getProgramParameter(program, gl.LINK_STATUS)) {

        let error = gl.getProgramInfoLog(shader);

        console.log('Failed to link program: ' + error);
        gl.deleteProgram(program);

        gl.deleteShader(fragmentShader);

        gl.deleteShader(vertexShader);

        return null;
    }

    gl.useProgram(program);

    gl.program = program;
}
```

怎么使用？

2、与shader有关联的库：（只需要我们去写vertex shader & fragment shader）

Threejs

Pixijs

...

以threejs的code为例：

```
<script type="x-shader/x-vertex" id="vertexshader">
    attribute float size;
    varying vec3 vColor;
    void main() {
        vColor = color;
        vec4 mvPosition = modelViewMatrix * vec4( position, 1.0 );
        gl_PointSize = size * ( 300.0 / -mvPosition.z );
        gl_Position = projectionMatrix * mvPosition;
    }
</script>

<script type="x-shader/x-fragment" id="fragmentsshader">
    uniform sampler2D texture;
    varying vec3 vColor;
    void main() {
        gl_FragColor = vec4( vColor, 1.0 );
        gl_FragColor = gl_FragColor * texture2D( texture, gl_PointCoord );
    }
</script>
```

```
var shaderMaterial = new THREE.ShaderMaterial( {
    uniforms: uniforms,
    vertexShader: document.getElementById( 'vertexshader' ).textContent,
    fragmentShader: document.getElementById( 'fragmentsshader' ).textContent,
    blending: THREE.AdditiveBlending,
    depthTest: false,
    transparent: true,
    vertexColors: true
} );
```

学习参考

工具&教程：

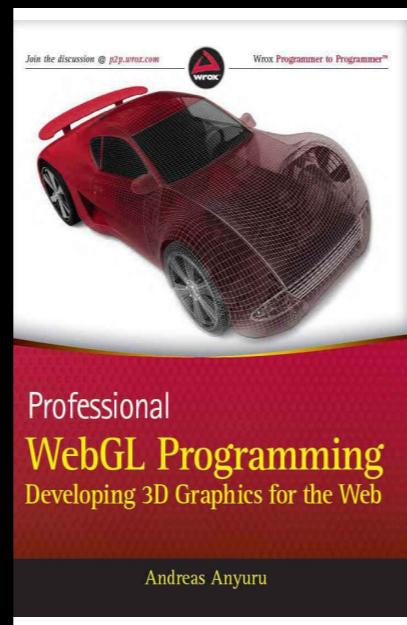
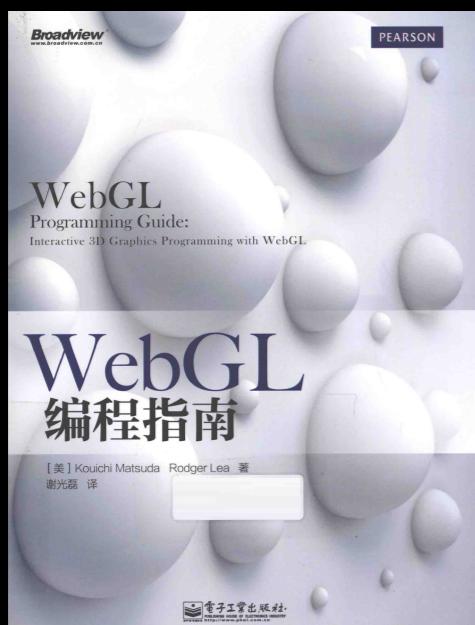
[shaderToy](#)

[WEBGL理论基础](#)

[The Book of Shaders](#)

[线性代数（英文版）](#)

书籍：



英文原版

Q & A