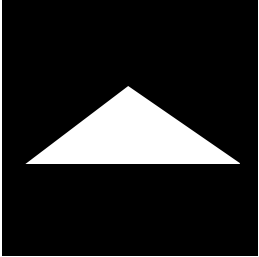
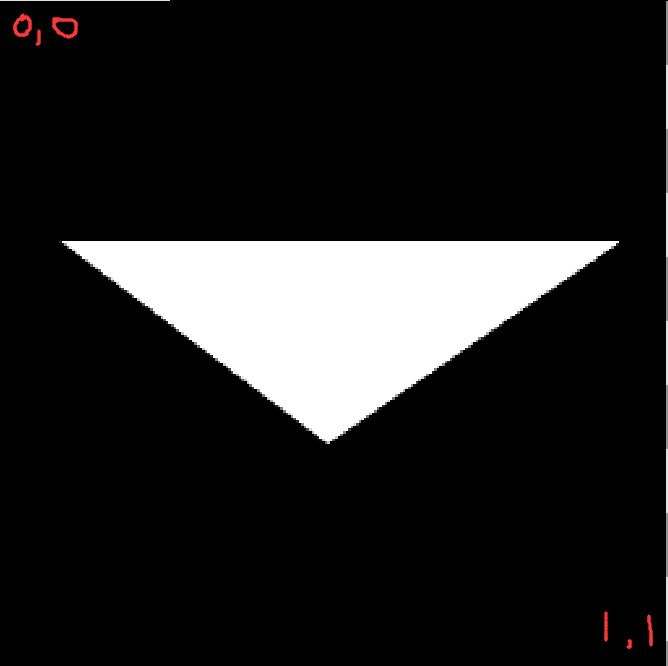
**Unity处理渲染平台差异的方式**

贴图资源



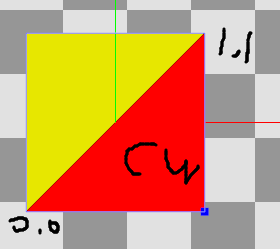
**DX**

1. 贴图UV

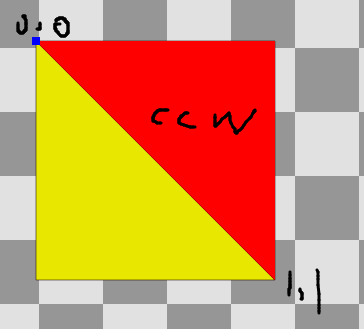


1. 普通流程

OS



CS



Clip

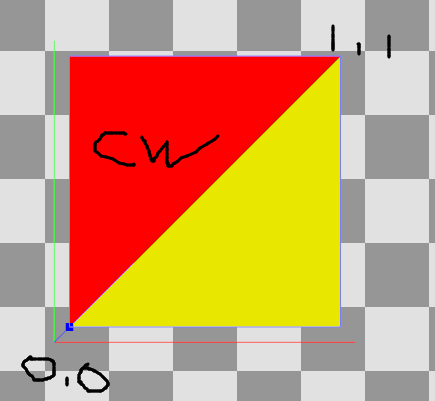


输出



1. Blit到FBO的流程

OS



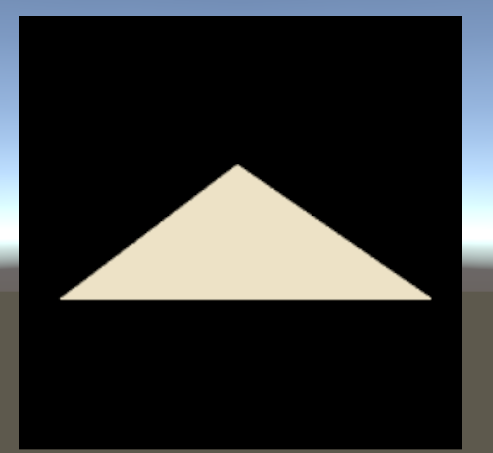
CS



Clip

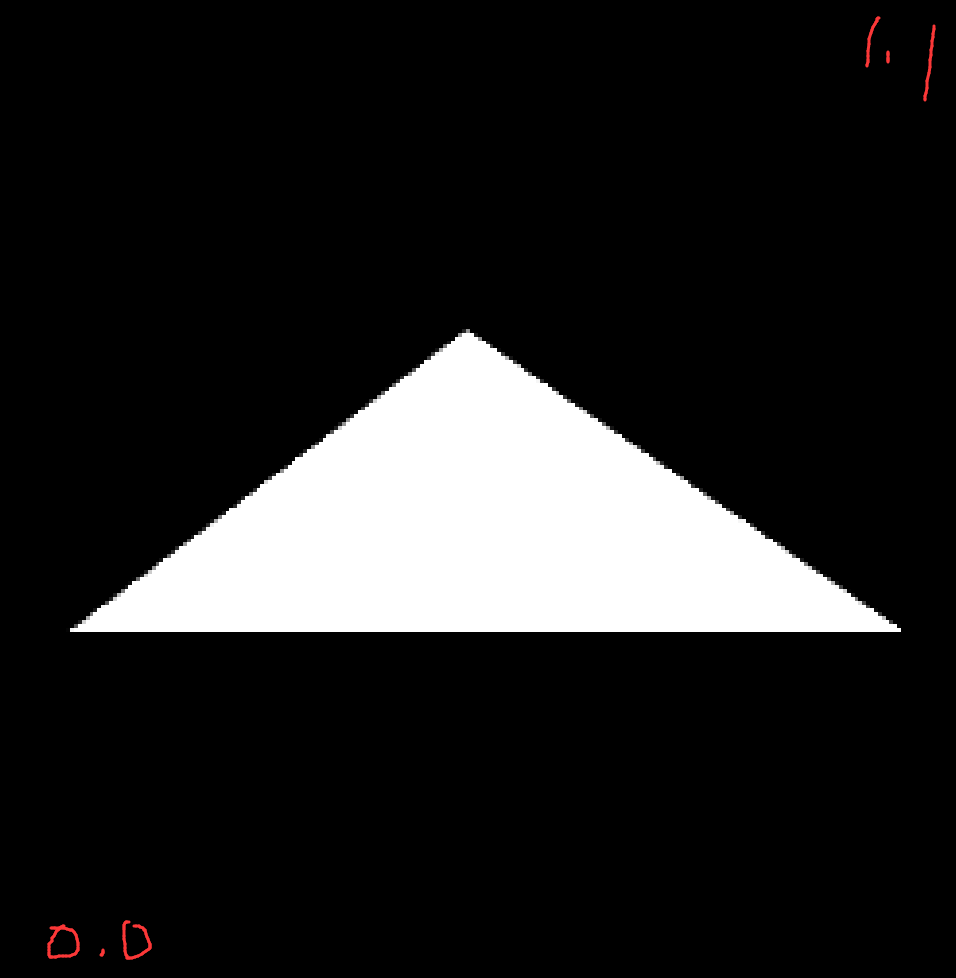


输出



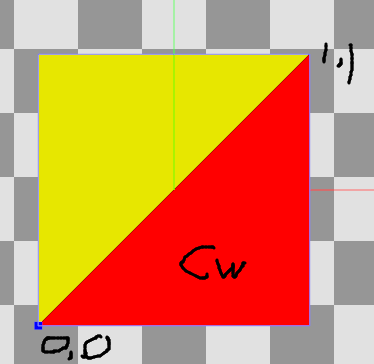
**Opengl es**

1. 贴图UV

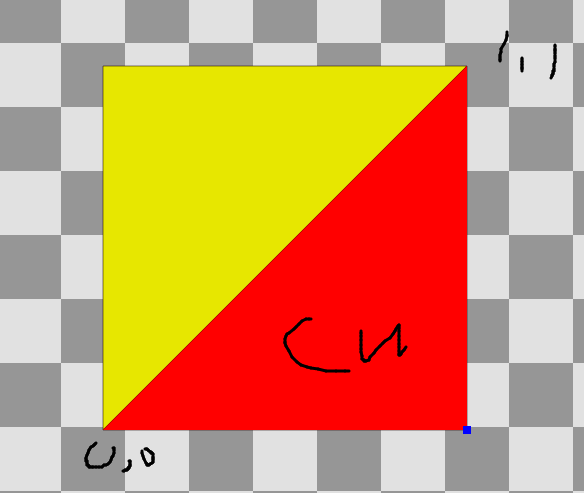


1. 普通流程

Os



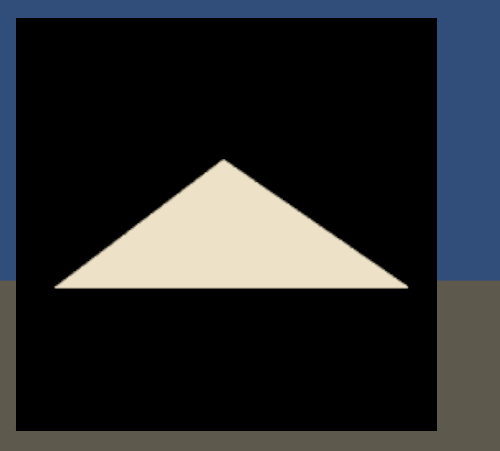
CS



Clip



输出



1. Blit到FBO的流程

Os

同DX

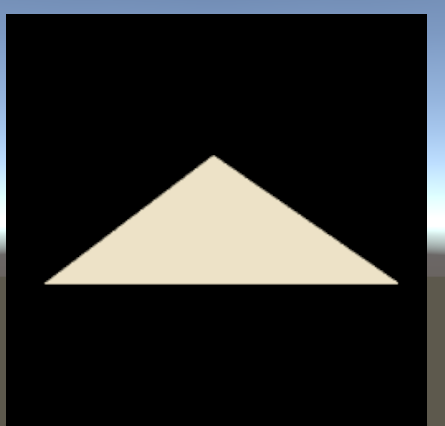
CS

同DX

Clip

同DX

输出



**总结：**

DX：

1. 纹理上下会颠倒，但是左上角为(0，0)，所以左下角uv为(0，0)的模型采样后不会颠倒。
2. 顶点着色器内，投影矩阵会翻转裁剪坐标系的y值，表现上为上下颠倒，三角形绕序改变，Unity设置CCW为正面（故开发时以顺时针为正面）。用\_ProjectionParams.x来判断投影矩阵是否翻转。
3. 从相机RT Blit到FBO时，因为RT的左上角为(0，0)，所以全屏幕的quad mesh的左下角会采样到RT的左上角，故颠倒RT，从而抵消前面的颠倒。

无论是纹理还是RT，都发生反转，以保证（0，0）在dx还是opengl采样结果都一样。

Opengl ES：

1. 纹理不会颠倒，而且左下角为(0，0)，所以左下角uv为(0，0)的模型采样后不会颠倒。
2. 顶点着色器内，投影矩阵不会翻转裁剪坐标系的y值，三角形绕序不改变，Unity设置CW为正面（故开发时以顺时针为正面）
3. 从相机RT Blit到FBO时，因为RT的左上下角为(0，0)，所以全屏幕的quad mesh的左下角会采样到RT的左下角，故不颠倒RT。