1. 认识shader的结构
2. shader命名：Shader"test/01myshader"

//这里指定Shader的名字，与文件名不要求保持一致(斜杠确定路径)

1. Properties属性：定义可在外部调节的变量，（！！！这个部分不可在段尾打上“；”）。

常用属性：

//属性名 （“外部显示名”，固定类型） = RGBA四元数

\_Color("Color",Color)=(1,1,1,1)

\_Vector("Vector",Vector)=(1,2,3,4)

\_Int("Int",Int)= 34234

\_Float("Float",Float) = 4.5

\_Range("Range",Range(1,11))=6

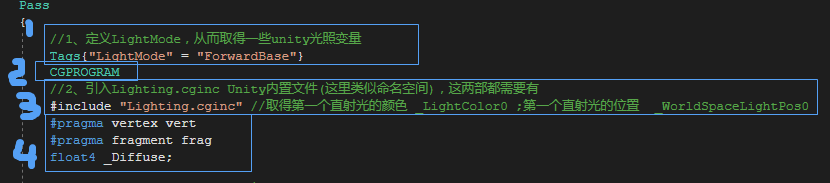
\_2D("Texture",2D) = "red"{}

\_Cube("Cube",Cube) = "white"{}//天空盒

\_3D("Texure",3D) = "black"{}

1. SubShader: 子Shader模块，可以有很多个，在不同的显卡实现不一样的效果，从第一个SubShader开始实现，如果第一个SubShader里面的效果有一些无法实现，会自动运行下一个SubShader，以此类推。（游戏在不同性能的硬件上运行时可能会使用不同的SubShader）
2. Pass: //Pass块，相当于方法，至少有一个Pass，可以有很多
3. 简单的光照模型

Pass中的处理步骤

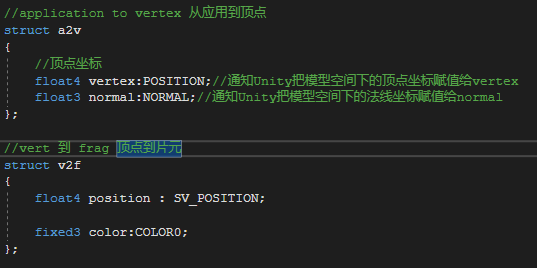


1,3的过程都可以理解成在c#脚本编写中的引用命名空间；

2，需要对应ENDCG

4，声明顶点、片元着色器；声明需要调用的外部变量

5、使用两个结构体来储存变换中的数据



**从应用程序传递到顶点函数的语义有哪些a2v：**

POSITION 顶点坐标（模型空间下的）//通知Unity把模型空间下的顶点坐标赋值给vertex

NORMAL 法线( 模型空间下) //通知Unity把模型空间下的法线坐标赋值给normal

TANGENT 切线（模型空间）

TEXCOORD0 ~ｎ　纹理坐标

COLOR 顶点颜色

**从顶点函数传递给片元函数的时候可以使用：v2f**

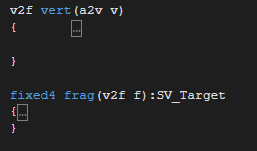
SV\_POSITION 剪裁空间中的顶点坐标（一般是系统直接使用）

COLOR0 可以传递一组值 4个

COLOR1 可以传递一组值 4个

TEXCOORD0~7 传递纹理坐标

6、编写两个控制顶点处理和片元处理的方法



UnityCG.cginc中一些常用的函数

//摄像机方向（视角方向）

float3 WorldSpaceViewDir(float4 v)

根据模型空间中的顶点坐标 得到 （世界空间）从这个点到摄像机的观察方向

float3 UnityWorldSpaceViewDir(float4 v)

世界空间中的顶点坐标==》世界空间从这个点到摄像机的观察方向

float3 ObjSpaceViewDir(float4 v)

模型空间中的顶点坐标==》模型空间从这个点到摄像机的观察方向

//光源方向

float3 WorldSpaceLightDir(float4 v)

模型空间中的顶点坐标==》世界空间中从这个点到光源的方向

float3 UnityWorldSpaceLightDir(float4 v)

世界空间中的顶点坐标==》世界空间中从这个点到光源的方向

float3 ObjSpaceLightDir(float4 v)

模型空间中的顶点坐标==》模型空间中从这个点到光源的方向

//方向转换

float3 UnityObjectToWorldNormal(float3 norm)

把法线方向 模型空间==》世界空间

float3 UnityObjectToWorldDir(float3 dir)

把方向 模型空间=》世界空间

float3 UnityWorldToObjectDir(float3 dir)

把方向 世界空间=》模型空间

normalize() 用来把一个向量，单位化（原来方向保持不变，长度变为1）

max() 用来取得函数中最大的一个

dot 用来取得两个向量的点积

\_WorldSpaceLightPos0 取得平行光的位置

\_LightColor0取得平行光的颜色

UNITY\_MATRIX\_MVP 这个矩阵用来把一个坐标从模型空间转换到剪裁空间

\_World2Object 这个矩阵用来把一个方向从世界空间转换到模型空间

UNITY\_LIGHTMODEL\_AMBIENT用来获取环境光

7.光照模型

标准光照模型

自发光；高光反射；

Blinn光照模型

Specular=直射光 \* pow( max(cosθ,0),10) θ:是反射光方向和视野方向的夹角；

Blinn-Phong光照模型

Specular=直射光 \* pow( max(cosθ,0),10) θ:是法线和x的夹角 x 是平行光和视野方向的平分线；

漫反射

Diffuse = 直射光颜色 \* max(0,cos夹角(光和法线 的夹角) ) cosθ = 光方向· 法线方向；

半兰伯特光照模型

Diffuse = 直射光颜色 \*（ cosθ \*0.5 +０.５ ）

环境光；

1. 抖动效果（使用数学函数）

<https://zh.numberempire.com/graphingcalculator.php>?

Y=Asin(Bx + C)

\_Time.xyzw;

X:0.5time;Y:1time;