Shader命令的语法为： Shader “name” { [Properties] Subshaders {…} [Fallback] }

Properties，着色器中有一个用来定义着色器属性的列表，任何定义在其中的属性都可以由开发人员在Unity的Inspector面板中编辑和调整，典型的属性包括颜色、纹理以及任何被着色器所使用的数据。基本语法为 Peoperties { Property [Property …] }

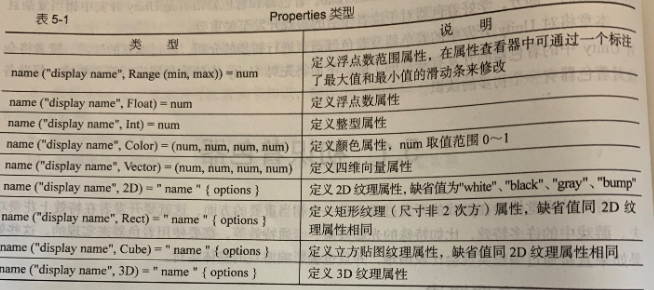
结构:

Properties {

\_RangeValue (“Range Value”, Range(0.1, 0.5)) = 0.3

变量名 属性面板显示名 变量类型 默认值

}



包含在纹理属性的大括号中的option是可选的：

TexGen 纹理生成模式，纹理自动生成纹理坐标时的模式。可以试ObjectLinear、EyeLinear、SphereMap、CubeReflect或CubeNormal，这些模式和OpenGL纹理生成模式相对应。注意如果使用自定义顶点片元着色器，那么纹理生成将被忽略。

LightmapMod 光照贴图模式，如果给出这个选项，纹理会被渲染器的光线贴图所影响，即纹理不能被应用在材质中，而是使用渲染器中的设定。

Subshader，着色器程序还包含一个子着色器列表，即Subshaders。子着色器列表是真正用来呈现渲染物体的部分。子着色器列表中有且至少有一个子着色器，即Subshader。当加载一个着色器程序时，Unity将遍历该列表来获得第一个能被用户机器支持的子着色器。

Subshader的基本语法为:

Subshader { [Tags] [CommonState] Passdef [Passdef …] }

Pass: 通道列表，能够选择是否为所有通道初始化所需要的通用状态。

当Unity 3D选择一个Subshader进行渲染的时候，将优先渲染一个被每个通道所定义的对象。并且有时在一些显卡上，所需要的效果不能通过单次通道来完成，就必须使用多次通道。定义通道类型有RegularPass、UserPass、GrabPass。

例子:

SubShader {

Tags { “Queue” = “Transparent” } *//渲染队列为透明队列*

Pass {

Lighting Off *//关闭光照*

SetTexture [\_MainTex] {} *//设置纹理*

}}

Subshader Tags, 子着色器使用标签来告诉渲染引擎何时渲染以及如何渲染对象，也就是说，Tags相对于向系统传递渲染信息的一个总指令。基本语法为:

Tags { “TagName1” = “Value1” “TagName2” = “Value2” }

常用的标签有三种类型:

Queue tag队列标签。可以决定对象被渲染的次序，也就是说，着色器通过对象所归属的渲染队列来保证物体先后渲染的顺序。并且任何透明物体都可以通过这种方法确保自身不在透明物体渲染之后渲染。

ShaderLab中有3种预定义的可选值:

Background(背景)：用于渲染天空盒之类的对象，对应值为1000。

Geometry(几何体): 不透明的几何体使用这个队列。此队列为默认选项，被用于大多数对象，对应值为2000。

AlphaTest(Alpha测试): 用于开启Alpha测试，对应值为2450。

Transparent(透明): 与Geometry相对，透明的物体使用这个队列。任何采用alpha混合的对象应该在这里渲染，比如玻璃 、粒子效果等，对应值为5000。

Overlay(覆盖): 此渲染队列被用于实现叠加效果。任何需要最后渲染的对象应该放置在此处，比如镜头光晕，对应值为4000。

#透明渲染队列为了达到最优的性能，优化了对象绘制次序，其他渲染队列根据距离来排序对象，从最远的对象开始，由远至近渲染。