着色器可以定义一个参数列表，可以由开发者在[材质检视面板](http://game.ceeger.com/Manual/Materials.html)编辑参数。着色器文件中的Properties块定义了这些参数：

语法：

**Properties** { *Property* [*Property ...*] }

定义属性块，其中可包含多个属性，其定义如下

*name* ("*display name*", **Range** (*min*, *max*)) = *number*

定义浮点数属性，在检视器中可通过一个标注最大最小值的滑条来修改。

*name* ("*display name*", **Color**) = (*number*,*number*,*number*,*number*)

定义颜色属性。

*name* ("*display name*", **2D**) = "*name*" { *options* }

定义2D纹理属性

*name* ("*display name*", **Rect**) = "*name*" { *options* }

定义长方形（非2次方）纹理属性

*name* ("*display name*", **Cube**) = "*name*" { *options* }

定义立方贴图纹理属性

*name* ("*display name*", **Float**) = *number*

定义浮点数属性

*name* ("*display name*", **Vector**) = (*number*,*number*,*number*,*number*)

定义四个向量组成的属性

细节：

包含在着色器中的每一个属性通过name索引（在Unity中, 通常使用下划线来开始一个着色器属性的名字）。属性会将display name显示在材质检视器中，还可以通过在等符号后为每个属性提供缺省值。

1.对于Range和Float类型的属性只能是单精度值。

2.对于Color和Vector类型的属性将包含4个由括号围住的数描述。

3.对于纹理(2D, Rect, Cube) 缺省值既可以是一个空字符串也可以是某个内置的缺省纹理："white", "black", "gray" or "bump" 。

例子：

**Properties   
{**

**// properties for water shader**

**// 水着色器的属性**

**\_WaveScale ("Wave scale", Range (0.02,0.15)) = 0.07 // sliders \_ReflDistort ("Reflection distort", Range (0,1.5)) = 0.5**

**\_RefrDistort ("Refraction distort", Range (0,1.5)) = 0.4**

**\_RefrColor ("Refraction color", Color) = (.34, .85, .92, 1) // color**

**\_ReflectionTex ("Environment Reflection", 2D) = "" {} // textures**

**\_RefractionTex ("Environment Refraction", 2D) = "" {}**

**\_Fresnel ("Fresnel (A) ", 2D) = "" {}**

**\_BumpMap ("Bumpmap (RGB) ", 2D) = "" {}**

**}**

Texture property options 纹理属性选项

包含在纹理属性的大括号中的选项是可选的。可能的选项有：

**TexGen** *texgenmode* 纹理生成类型

纹理的自动生成纹理坐标时的模式。可以是ObjectLinear, EyeLinear, SphereMap, CubeReflect, CubeNormal的其中之一;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程序，那么纹理生成将被忽略。

**LightmapMode** 光照贴图模式

如果给出这个选项，纹理将能被渲染器的光线贴图属性所影响。纹理不能被使用在材质中，而是取自渲染器的设定。请参考[渲染器脚本文档](http://game.ceeger.com/Script/Renderer/Renderer.lightmapIndex.html)。

例子：

// EyeLinear texgen mode example  
// EyeLinear 纹理生成模式  
Shader "Texgen/Eye Linear" {  
    Properties {  
        \_MainTex ("Base", 2D) = "white" { TexGen EyeLinear }  
    }  
    SubShader {  
        Pass {  
            SetTexture [\_MainTex] { combine texture }

            // 注意如果使用自定义顶点程序，那么纹理生成将被忽略。

            // Lighting On  
        }  
    }  
}

http://blog.sina.com.cn/s/blog\_89d90b7c01019pnc.html