# **UnityShader-模仿炉石传说卡牌动态效果(完)**

作者：[枸杞忧天](http://gad.qq.com/user/index?id=1075523)

链接：<http://gad.qq.com/article/detail/286352>

在上一部分[《UnityShader-模仿炉石传说卡牌动态效果(二)》](http://gad.qq.com/article/detail/286351)中，鄙人对“移动”、“旋转”效果的实现及两种混合效果的显示差异进行了总结。

这第三部分(最终部分)会对自定义材质编辑器作出介绍，并对全文作出总结。

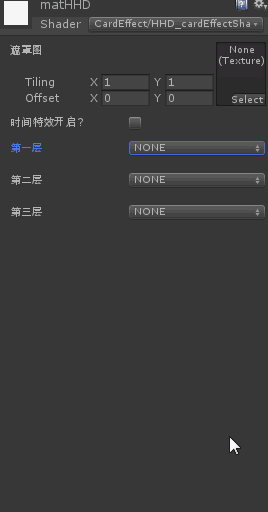
那就开车吧，嘀嘀。

**自定义材质编辑器**

**自定义材质编辑器**，就是根据自身需求，调整Inspector面板上的显示，以方便美术、特效同学更快的配置材质。当然，最主要的是让美术、特效妹子觉得你这个人 -- **专业**。



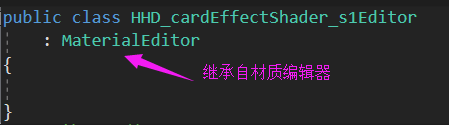
↑ **没有**使用自定义材质编辑器



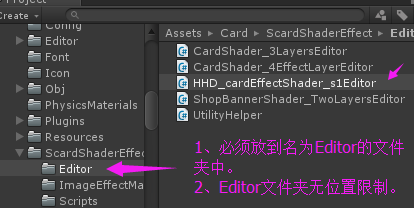
↑ **使用**了自定义材质编辑器

**开始使用自定义材质编辑器**

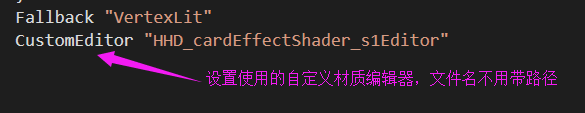
**◇ 创建Editor.cs文件并与Editor扯上关系**



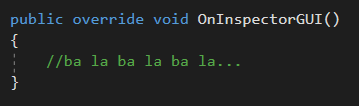
**◇ 将文件放到Editor文件夹中**



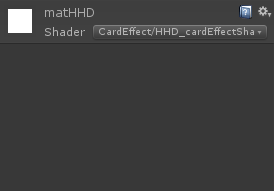
**◇ 在Shader中指定使用的自定义编辑器文件**



**◇ 在Editor文件中重写OnInspectorGUI函数**



注意：自定义编辑器的布局都发生在OnInspectorGUI函数中，如果你重写了这个函数，并且没有调用父类的OnInspectorGUI，那么材质Inspector界面将变得一片空白。



↑ 一片空白的Inspector面板

**自定义材质编辑器的书写套路**

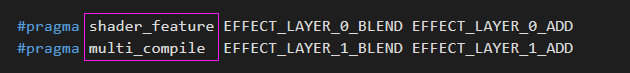
其实，编辑材质，主要就是编辑两项内容：**Properties** 和 **Keywords**。

**Properties**

“Properties - 属性”是指一个Shader文件的Properties部分，Properties可以通过Inspector面板设置并保存在**.mat**文件，在后续渲染中mat文件在每一次**Draw**中将保存的数值传递给**shader**。

**Keywords**

“Keywords - 关键词”是指定义在一个Shader文件中的关键词，通常这么定义：



这种关键词的意义是：可以在保持**一份**Unity Shader Lab文件的基础上，达到编译出多个.shader(GPU读取的shader)文件的效果(**多重编译**)。

比如：有一个特效Shader需要支持两层特效，但是再实际使用时可能用一层或者用两层。那如果不用“多重编译”，那么用支持两层特效的shader去处理一层特效时，是一种浪费。

而设置了Keywords后，Unity会根据Keywords和多重编译的方式（**multi\_compile**和**shader\_feature**）生成多份shader。而通过在材质中设置Keyword，即可确定使用到的是具体的哪个.shader，真是**美滋滋**。



具体到**shader\_feature**和**multi\_compile**的差异可以看官方文档(搜关键词“**Making multiple shader program variants”**)，在这里就不再赘述了。

**OnInspectorGUI函数中的处理逻辑**

**获取shader中的Properties和Keywords**

**↓**

**查看是否包含某个keyword并根据keyword调整GUI以及属性的显示**

**↓**

**显示属性并检测属性修改**

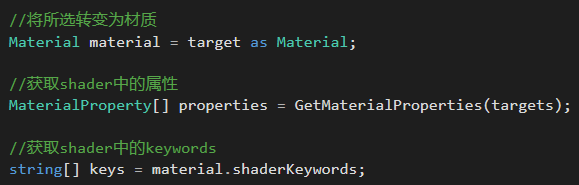
**↓**

**为材质设置新的keyword**

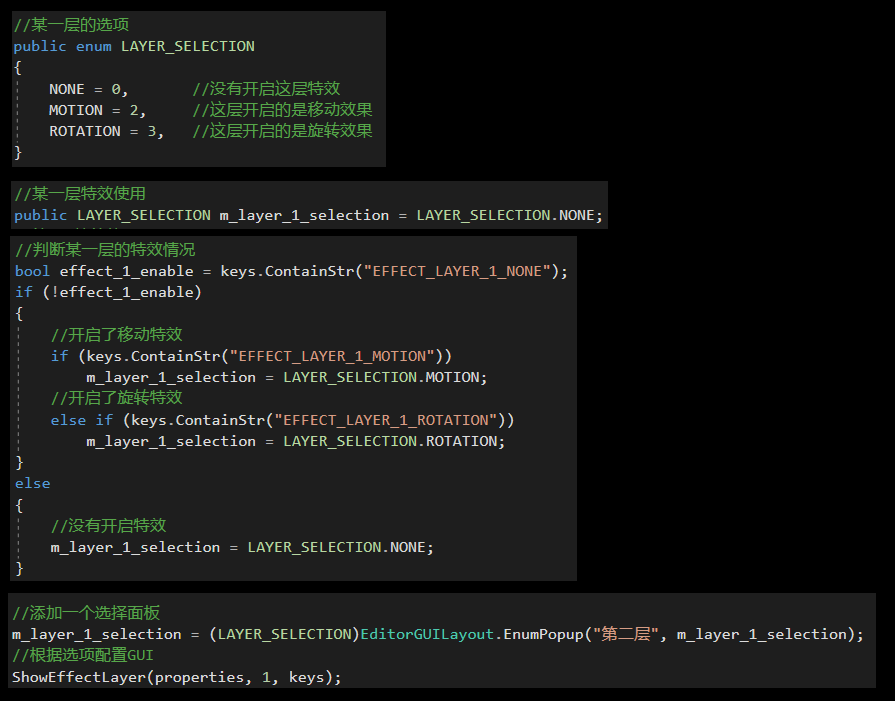
**↓**

**标记当前材质发生了变化**

**◇ 获取shader中的Properties和Keywords**

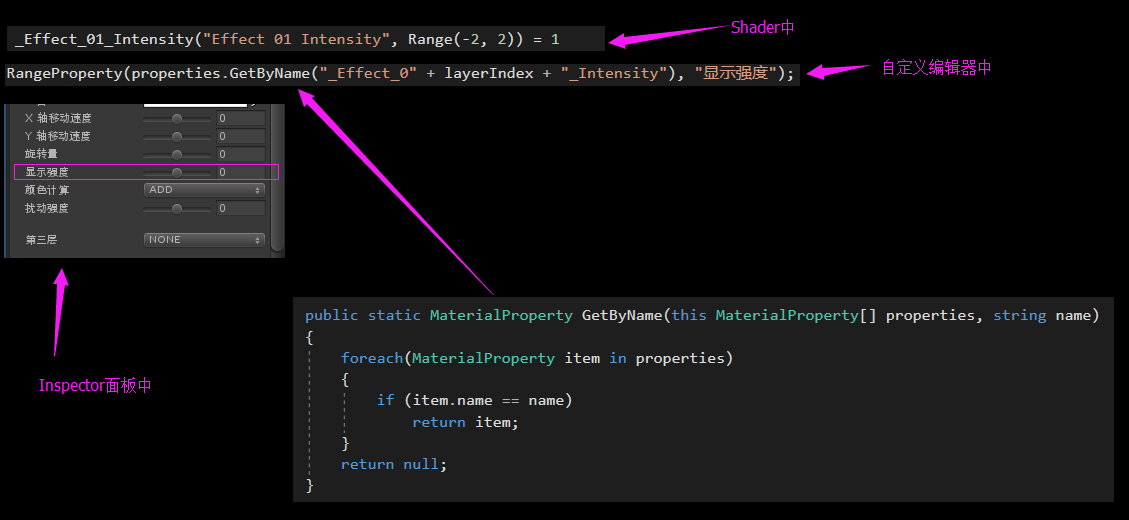


**◇ 查看是否包含某个keyword并根据keyword调整GUI以及属性的显示**



**◇ 显示属性并更新属性的修改**

因为已经得到了MaterialProperty，这个事情就简单多了，对照着Shader中的Properties做就行了，比如：



**为材质设置新的keyword 并 标记当前材质发生了变化**



**常用的材质属性修改API**

public Color **ColorProperty**(MaterialProperty **prop**, string **label**); //设置颜色

public float **FloatProperty**(MaterialProperty **prop**, string **label**); //设置浮点数

public Vector4 **VectorProperty**(MaterialProperty **prop**, string **label**); //设置向量

public Texture **TextureProperty**(MaterialProperty **prop**, string **label**); //设置纹理

...

更多API，在官方文档中搜索**MaterialEditor**即可。

至此，此次小总结就告一段落了，很多地方碍于鄙人目前的能力水平，可能没讲明白，多包涵...

**写在最后**

◇ 鄙人的思路来自一个YouTube上的视频，讲的实在是太tm清楚了，如果有时间的话建议你一定要看：

**https://www.youtube.com/watch?v=OYjMnMZe1Vg**

◇ 我们可以凭借图像渲染的知识和对Unity Shader Lab的掌握，制作更多的材质编辑工具，实现更多的渲染效果，并不断提升渲染质量以及提高渲染效率。

但是，设计更好、更炫的特效，**还是得看美术和特效妹子，**她们才是最重要的。

◇ 此项目总结的经验和截图素材来自鄙人参与的TCG卡牌游戏**《三国卡牌大战》**。

附上游戏官网：**http://www.sgscard.cn/**

全文完。