# Nexus Engine材质系统设计

## 从项目需求的角度分析

* 常用的点光源，方向光，聚光灯渲染；
* 常用的非光照模式渲染；
* 可能用到的：当房子等对象遮挡摄像机时，房子变成半透明；
* 可能用到的：当主角被遮挡时，多渲染一个zfail的pass；
* 可能用到的：按住alt键时，突出显示场景中所有可能交互的对象；可以使所有可交互对象增加一个zfail的pass；
* 鼠标指向对象时，显示选中状态；
* 任务或者什么事件刷出NPC、怪物时，NPC、怪物渐变出现；

## 材质系统设计概要

根据项目的需求，确定材质必须支持Multi-Pass渲染，这样上层可以根据游戏逻辑的需要开启、关闭某个Pass来实现上述功能。但是，问题是渲染底层为了渲染多光源，也是用了Multi-Pass结构，两者如何结合是一个最大的问题。为了解决这个问题，首先是将底层渲染各个Pass所需要的Drawing Policy暴露给了上层，这样虽然增加了系统之间的耦合度，但是也加强了材质对渲染的控制；其次在标准的渲染流程完成之后，增加了一个Drawing Extend扩展渲染，用来渲染材质需要的额外的Pass。

我们把**材质Shader+一组渲染状态**定义为一个Material Tech（这里避免了使用Pass这个单词，以和底层做一个概念上的区分）；我们把一些**Material Tech的组合**定义为一个Material对象，Material还需要管理所有Tech需要的Shader参数值，以及Material Tech与Drawing Policy之间的绑定关系（底层在渲染某个Policy的时候使用哪个Material Tech）。



## Actor系统结合

考虑到上述项目需求，一个场景中的Actor可能需要单独控制自己材质的某些参数值变化，所以Actor需要管理自己独有的材质。所以在static mesh component等组件中增加了克隆资源材质的功能。

## 杂项

1. 材质应该提供更多信息，例如在计算light map时alpha mask使用的贴图等；