* **NoiseMaterialManager：**

1. (2015.8.20) 管理材质，有名字或者Index两种访问方式，名字用std::string。最主要使用UINT类型的Index来访问纹理，NoiseMesh里面最终也是设置通过设定ID来设置纹理。若某个三角形没有设置纹理，则在shader里面的gIsXXXMapValid会设为0，关闭此三角形Texture mapping。
2. (2015.8.22)Mat Mgr的第一个元素（下标是0）是在构造函数中就创建的默认材质。
3. (2015.8.22)Material包含了基本颜色和贴图ID。然后在N\_Material里面有一个结构是N\_Material\_Basic，这个是一些基本数值的设置，然后N\_Material里面各种贴图ID就顾名思义了，可以用Noise Texture Manager的GetIndexByName来获取Tex ID。
4. (2015.8.22)没有无效的Material，如果哪个类设置了无效的mat ID就要把它的mat ID设为0（default）。（贴图可以不要，但是材质必须要有）。无效贴图的tex ID为-1。
5. (2016.2.22)统一使用std::string的名字来标识material，为了加快材质的查找和访问速度，特地添加了一个std::unordered<std::string,UINT>的哈希表，把material名字映射到对应material在vector里的index。一开始有想过只允许用string来访问material（即string-N\_Material的pair），但是如果要批量操作好像还是有点烦，因为并没有实现迭代器，所以还是允许用index对material进行访问。index是会随着资源的增加和删除而改变，所以index是不靠谱的，要用唯一标识符(uid)，这里我选择使用名字作为唯一标识符。Noise Texture Mgr也使用类似的，有个哈希表，用名字作为uid管理资源。
6. (2016.2.22)默认的材质不再放在index 0的位置，那样会显得有点奇怪。因为要使用material的对象要记录的只是material 的名字，所以检查是发生在在渲染之前，无效material会被替换成默认的material。