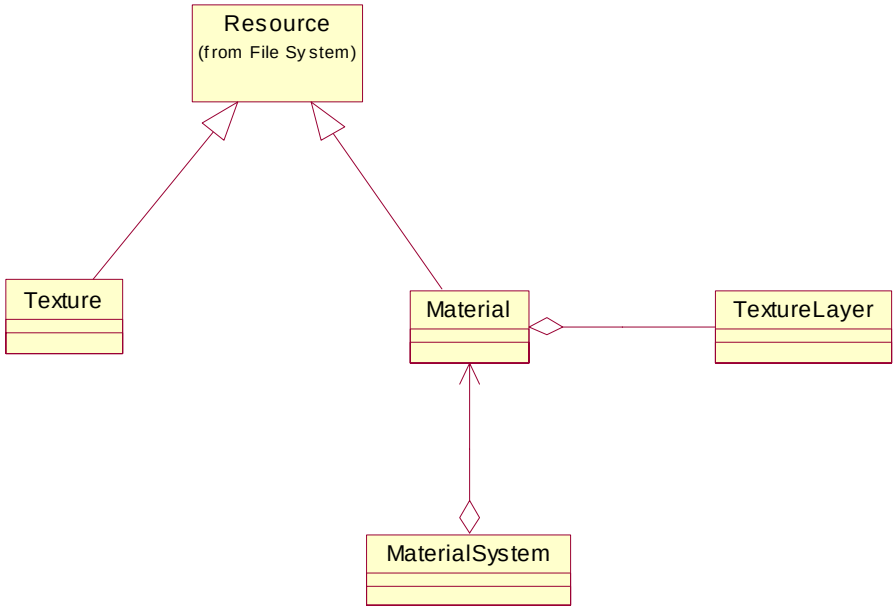
Ogre Material System分析

盛崇山
http://antsam.blogone.net
AntsamCGD@hotmail.com

这篇文章我们主要分析 Ogre Material System 的设计方法，同时也分析脚本同 Material System 的结合方式。在分析 Material System 之前我想，我们先分析 Ogre 的脚本系统的实现。下面看一下 Material System 相关类的 UML 图：



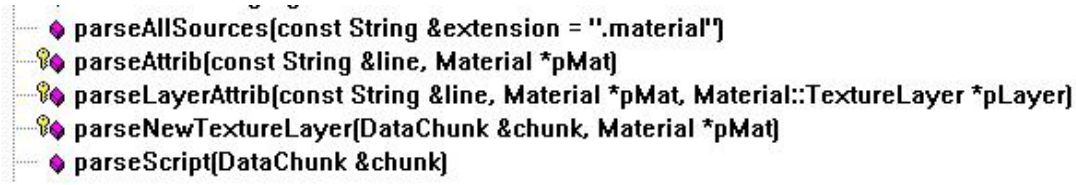
Material 是通过脚本读取的，现在我们看一个实例：

```
PlayPen/EnvmapPlanar
{
    ambient 1.0 1.0 1.0
    diffuse 1.0 1.0 1.0
    // Texture layer 0
    {
        texture RustySteel.jpg
    }
    // Texture layer 1
    {
        texture spheremap.png
        colour_op add
        env_map planar
    }
}
```

```
PlayPen/EnvmapSphere
{
	ambient 1.0 1.0 1.0
	diffuse 1.0 1.0 1.0
	// Texture layer 0
	{
		texture RustySteel.jpg
	}
	// Texture layer 1
	{
		texture Chrome.jpg
		colour_op add
		env_map spherical
	}
}
```

那我们什么时候来分析这些脚本的那？

在 Material System 中有几个函数：



先看第一个函数：

```
void MaterialManager::parseAllSources(const String& extension)
{
	StringVector materialFiles;
	DataChunk* pChunk;

	std::vector<ArchiveEx*>::iterator i = mVFS.begin();

	// Specific archives
	for (; i != mVFS.end(); ++i)
	{
		materialFiles = (*i)->getAllNamesLike( "./", extension);
		for (StringVector::iterator si = materialFiles.begin(); si != materialFiles.end(); ++si)
		{
			SDDataChunk dat; pChunk = &dat;
			(*i)->fileRead(si[0], &pChunk );
			LogManager::getSingleton().logMessage("Parsing material script: " + si[0]);
			parseScript(dat);
		}
```

```
        }
        // search common archives
        for (i = mCommonVFS.begin(); i != mCommonVFS.end(); ++i)
        {
            materialFiles = (*i)->getAllNamesLike( "./", extension);
            for (StringVector::iterator si = materialFiles.begin(); si != materialFiles.end(); ++si)
            {
                SDDataChunk dat; pChunk = &dat;
                (*i)->fileRead(si[0], &pChunk );
                LogManager::getSingleton().logMessage("Parsing material script: " + si[0]);
                parseScript(dat);
            }
        }
}
```

代码中用蓝色标志的代码作用依次是：

- 寻找后缀是extension的文件
- 读取文件数据
- 分析数据

好现在我们看第二个parseScript的代码：

```
void MaterialManager::parseScript(DataChunk& chunk)
{
    String line;
    Material* pMat;
    char tempBuf[512];

    pMat = 0;

    while(!chunk.isEOF())
    {
        line = chunk.getLine();
        // Ignore comments & blanks
        if (!(line.length() == 0 || line.substr(0,2) == "//"))
        {
            if (pMat == 0)
            {
                // No current material
                // So first valid data should be a material name
                // NB defer loading until later
                pMat = (Material*)createDeferred(line);
                // Skip to and over next {
                //  这里其实读取Material的名字
                chunk.readUpTo(tempBuf, 511, "{");
```

```
                }
                else
                {
                    // Already in a material
                    if (line == "}")
                    {
                        // Finished material
                        pMat = 0;
                    }
                    else if (line == "{")
                    {
                        // new pass
                        parseNewTextureLayer(chunk, pMat);
                    }
                    else
                    {
                        // Attribute
                        parseAttrib(line.toLowerCase(), pMat);
                    }
                }
            }
        }
    }
```

从上面代码可以看出Material的组织方式，可以对照前面列出的Material的例子。

更具体的代码可以自己查看，代码中为分析每个属性都定义了一个函数，而每个函数根据string的的值设置Material的相应值。