http://www.tairan.com/archives/6126/

**初学者教程**

 [第一课：新建一个窗口](http://www.tairan.com/archives/5583)

 [第二课： 画第一个三角形](http://www.tairan.com/archives/5595)

 [第三课： 矩阵](http://www.tairan.com/archives/5951)

 [第四课：彩色立方体](http://www.tairan.com/archives/5954)

 [第五课：纹理立方体](http://www.tairan.com/archives/5958)

 [第六课：键盘和鼠标](http://www.tairan.com/archives/5962)

 [第七课：模型加载](http://www.tairan.com/archives/5964)

 [第八课：基础光照模型](http://www.tairan.com/archives/5966)

**中级教程**

 [第九课：VBO索引](http://www.tairan.com/archives/5968)

 [第十课：透明](http://www.tairan.com/archives/5970)

 [第十一课：2D文本](http://www.tairan.com/archives/5972)

 [第十二课：OpenGL扩展](http://www.tairan.com/archives/5974)

 [第十三课：法线贴图](http://www.tairan.com/archives/5976)

 [第十四课：渲染到纹理](http://www.tairan.com/archives/5978)

 [第十五课：光照贴图](http://www.tairan.com/archives/5980)

 [第十六课：阴影贴图](http://www.tairan.com/archives/5982)

**1 绘制三角形**

void GL301::setRC()

{

cout << "301 setRC" << endl;

glGenBuffersARB(1, &vboId);

glBindBufferARB(GL\_ARRAY\_BUFFER, vboId);

glBufferDataARB(GL\_ARRAY\_BUFFER, sizeof(GL\_FLOAT)\* sizeof(vertices), vertices, GL\_STATIC\_DRAW);

}

void GL301::render()

{

cout << "301 render" << endl;

glBindBufferARB(GL\_ARRAY\_BUFFER, vboId);

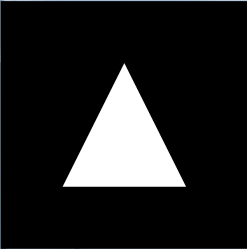
**glEnableVertexAttribArray(0);**

**glVertexAttribPointer(0, 3, GL\_FLOAT, GL\_FALSE, 0, NULL);**

glDrawElements(GL\_TRIANGLES, 3, GL\_UNSIGNED\_BYTE, indices);

glBindBufferARB(GL\_ARRAY\_BUFFER, NULL);

}



注意上面红色的部分，表示我们需要使用shader，实际上它的功能和glEnableClientStatus glVertexPointer是一样的（已经验证）

着色器变成使用glsl，它是opengl的一部分，与c和java不同，glsl必须在运行时编译，这意味着每次启动程序，所有的着色器将重新编译。

注意：和缓冲区一样，着色器不能直接访问：我们仅仅有一个编号（ID），真正的实现隐藏在驱动程序中