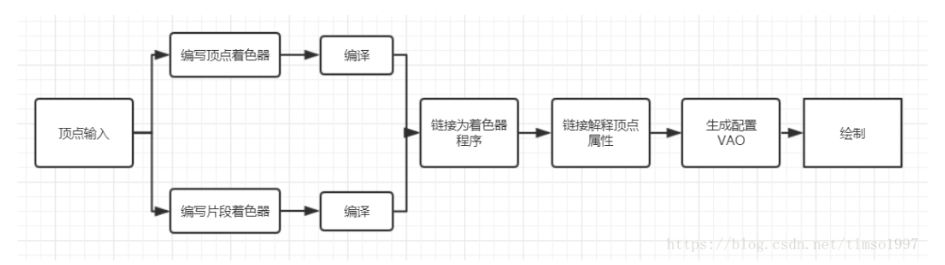


**实现思路:**





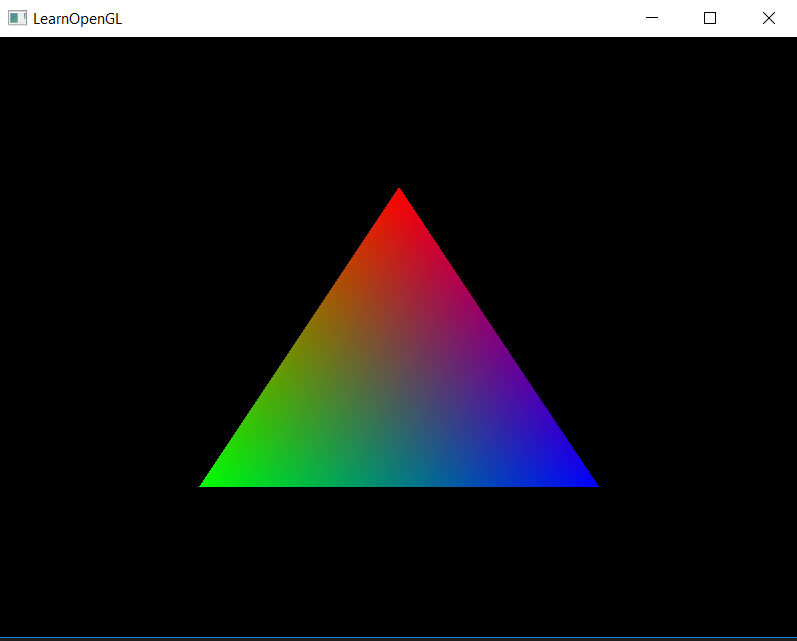










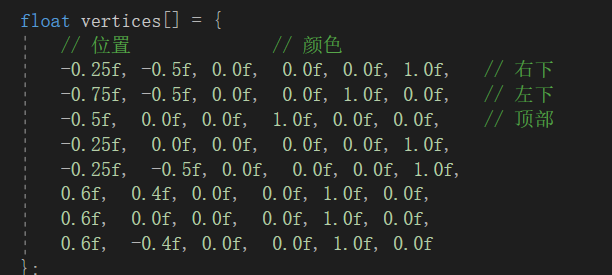


解释: 三种颜色均匀分布在三角形上的这种现象, 是因为片段着色器在渲染这个三角形时, 在光栅化阶段除了所指定的三个节点外, 还会生成额外的片段, 然后光栅会根据片段在三角形形状上的相对位置决定他们的插值位置. 而因为这是个正三角形, 所以三种颜色在三角形中进行等比例插值, 最终形成了三种颜色均匀分布的现象.

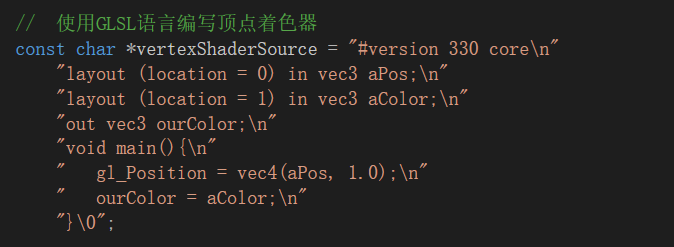
**实现思路:**

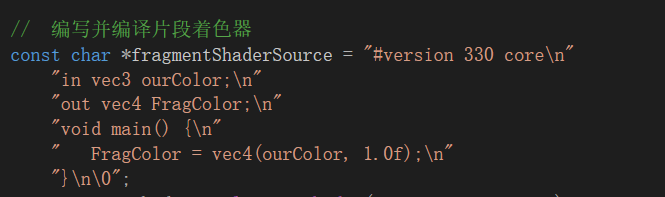
在三角形的基础上,

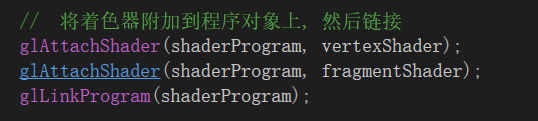
1. 在顶点数组中添加颜色属性



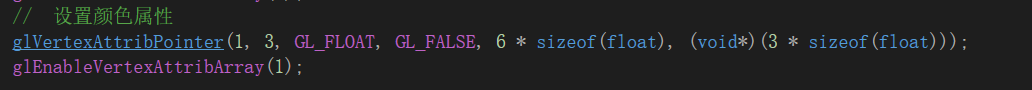
1. 重写着色器, 并链接新的着色器







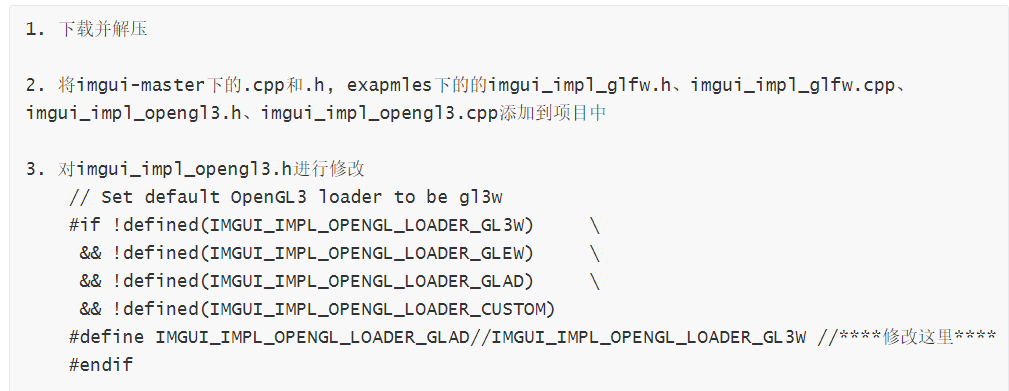
1. 使用glVertexAttribPointe设置顶点的颜色属性



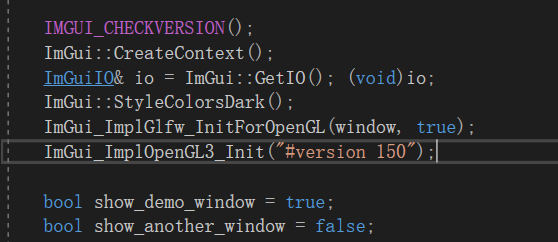


**实现思路:**

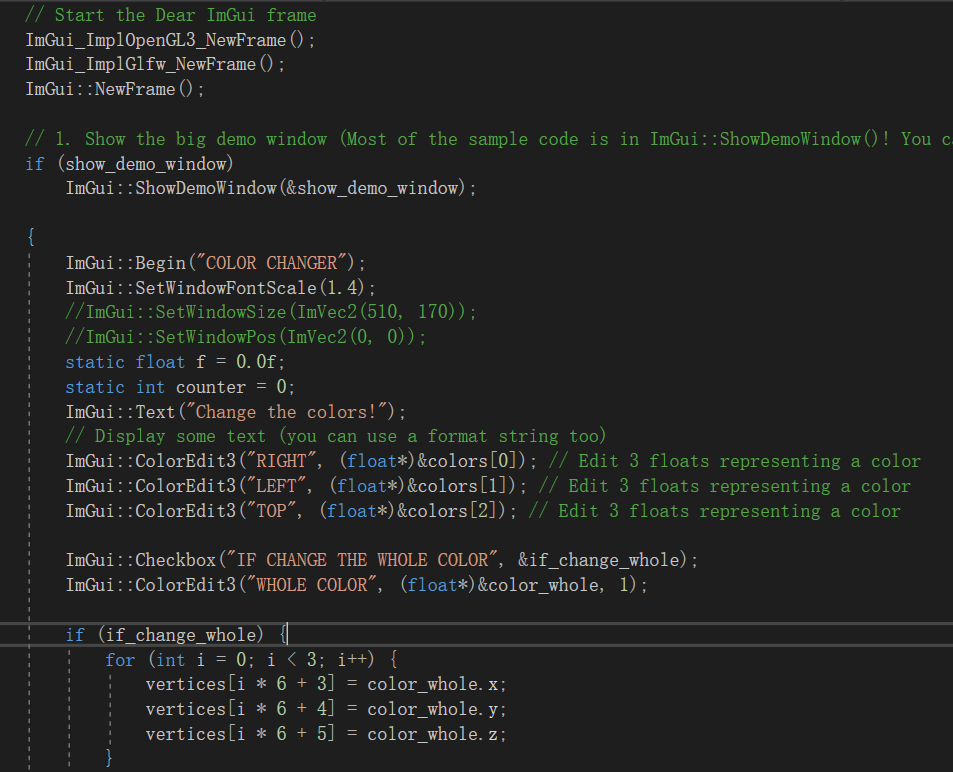
1. 配置imgUI.

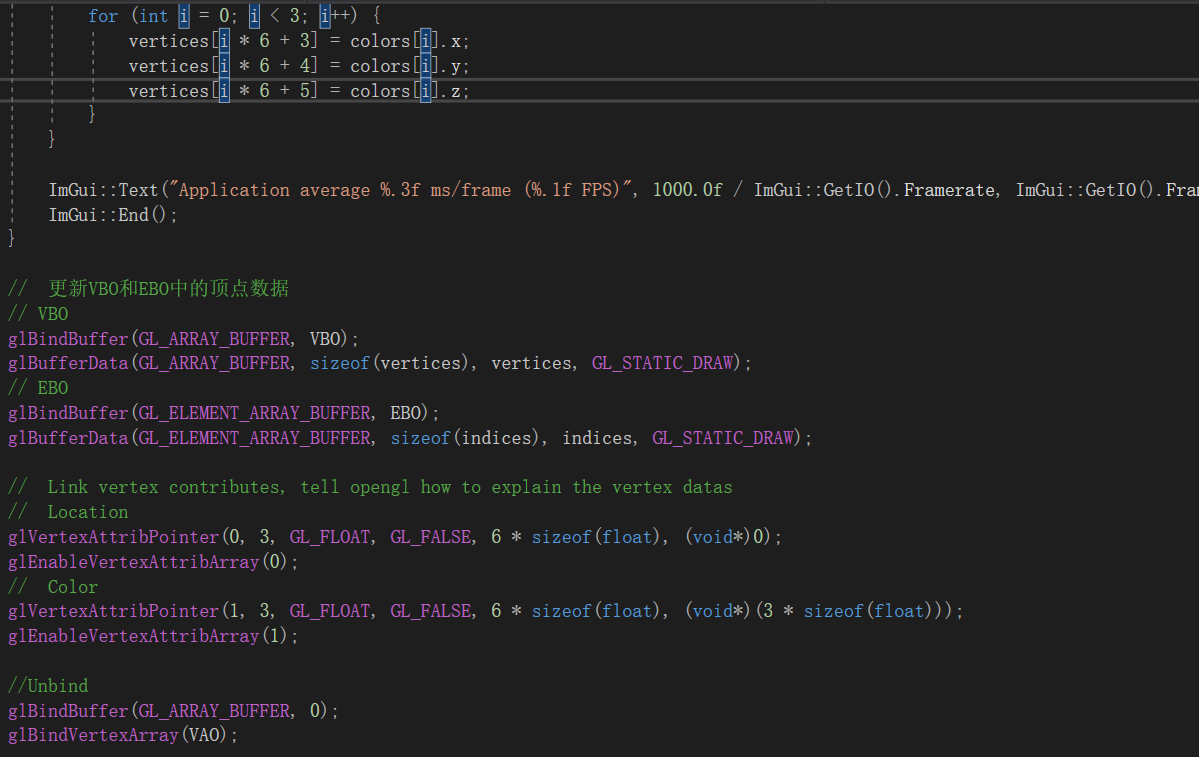


1. 初始化imgUI



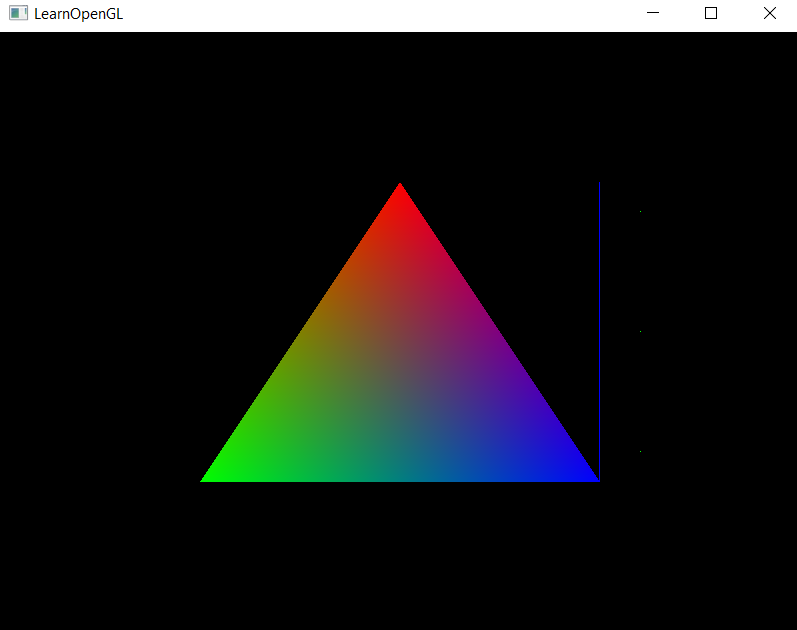
1. 设置面板, 并将顶点更新后的颜色放入缓冲中





**Bonus:**





从左至右分别是三角形, 直线, 三个绿色的点

**实现思路:**

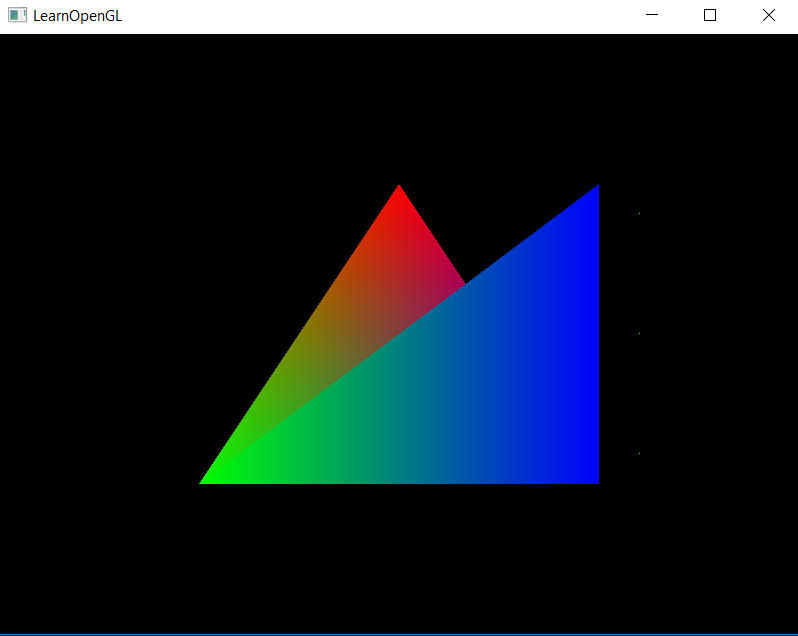
1. 点



1. 线

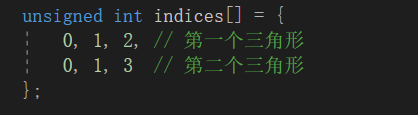




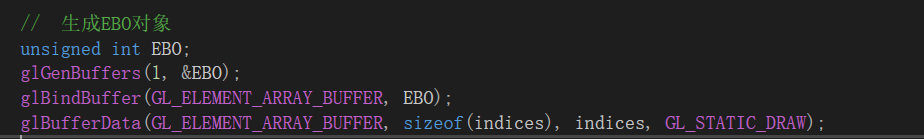


**实现思路:**

1. 定义顶点索引数组



1. 生成EBO对象



1. 通过函数glDrawElements进行渲染

