Open GL 在Mac 上的配置

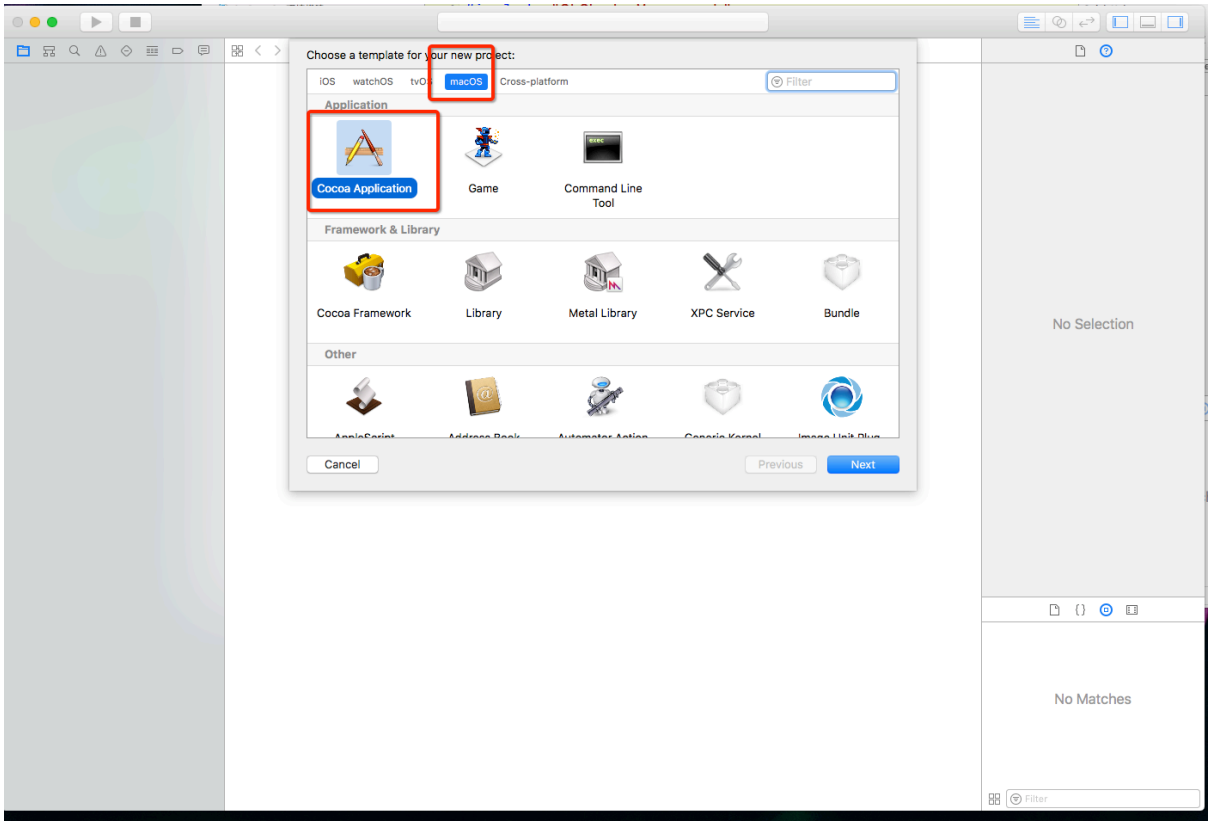
准备资源

- CLTools
- glew
- libGLTools.a

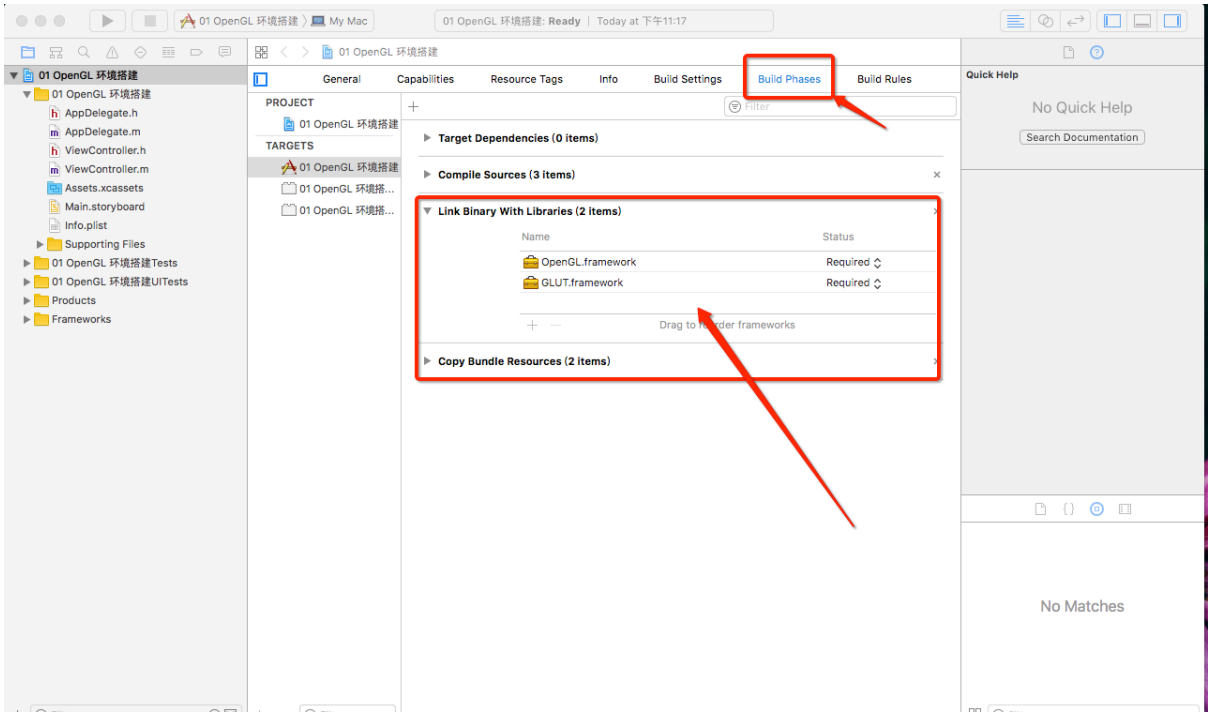
百度云盘分享地址：链接:http://pan.baidu.com/s/1i4PTEb7 密码:nbgc

现在开始配置 openGL 环境

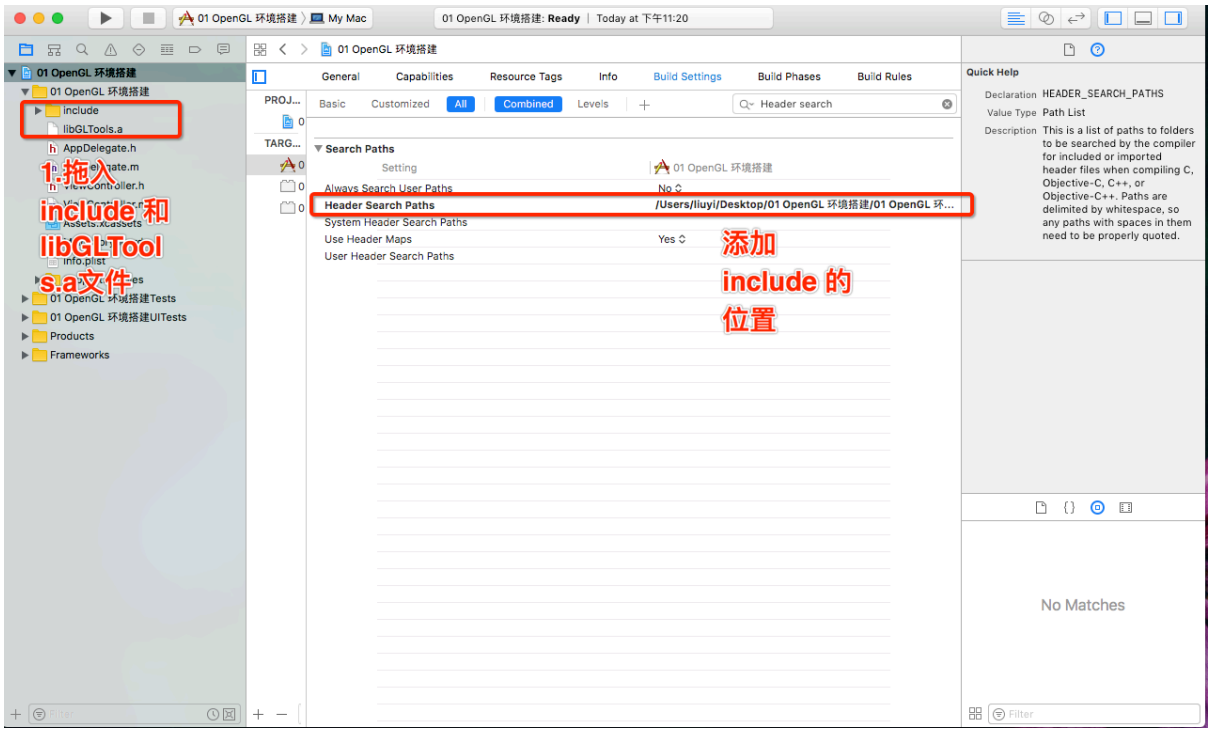
- 打开Xcode -> macOS -> Cocoa Application



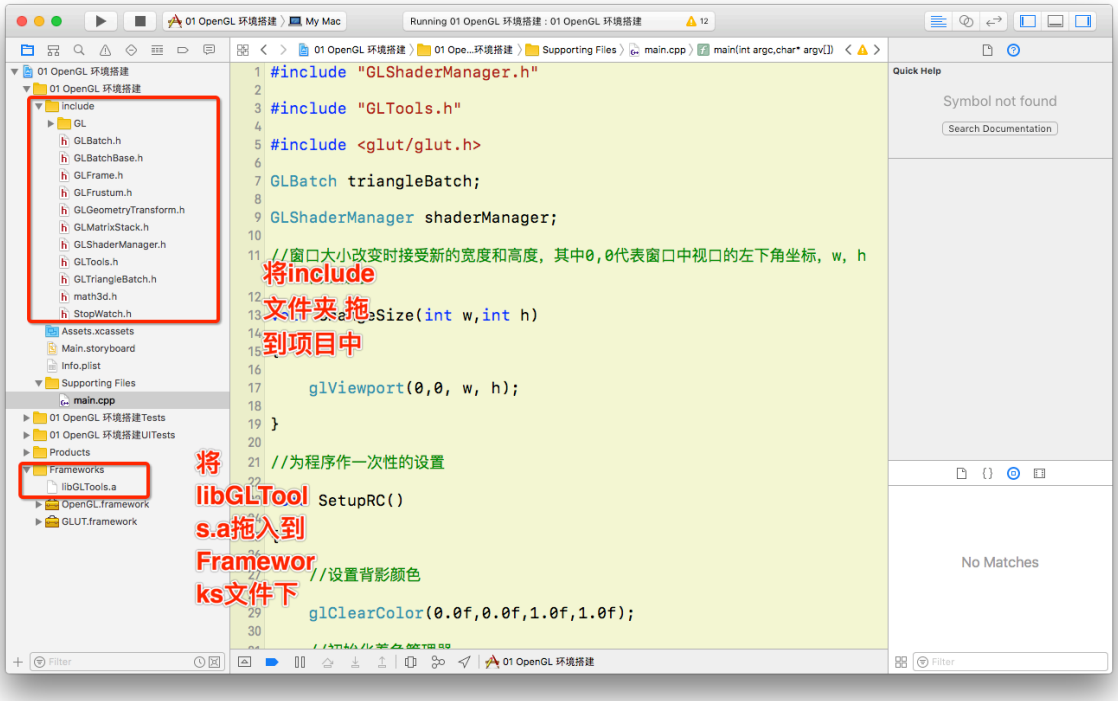
- 添加OpenGl.framework 和 GLUT.framework 两个系统库

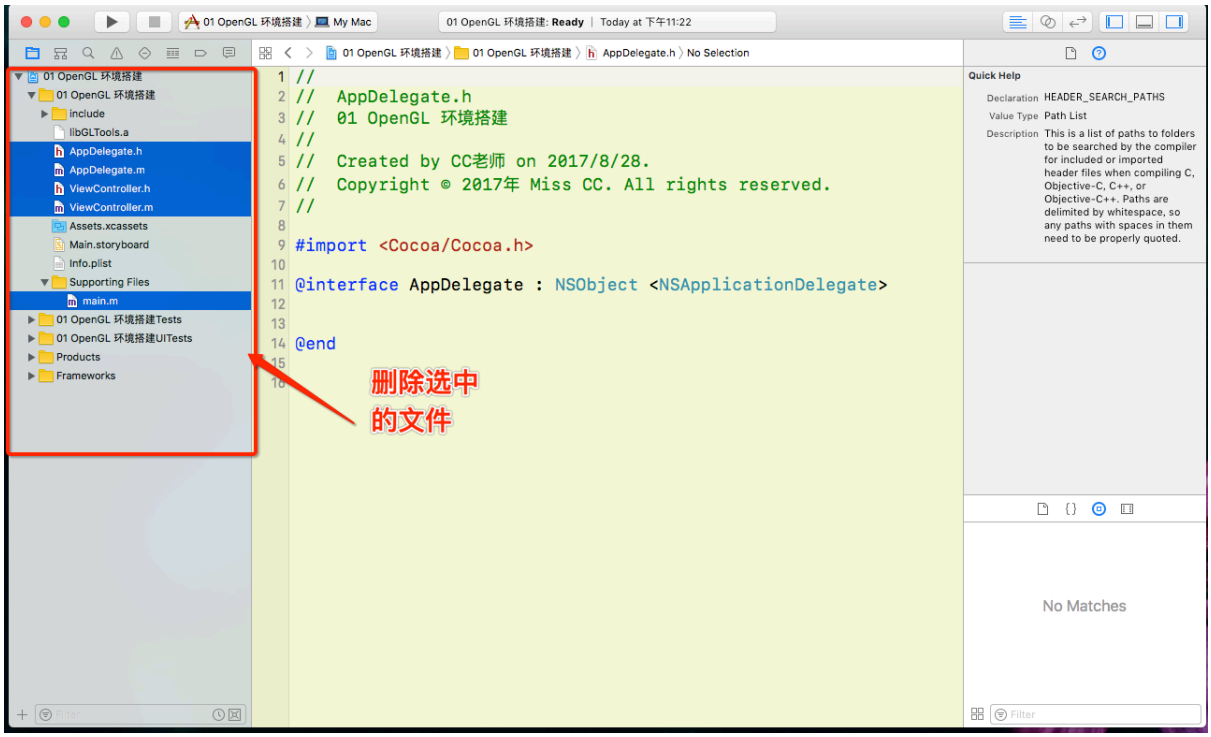


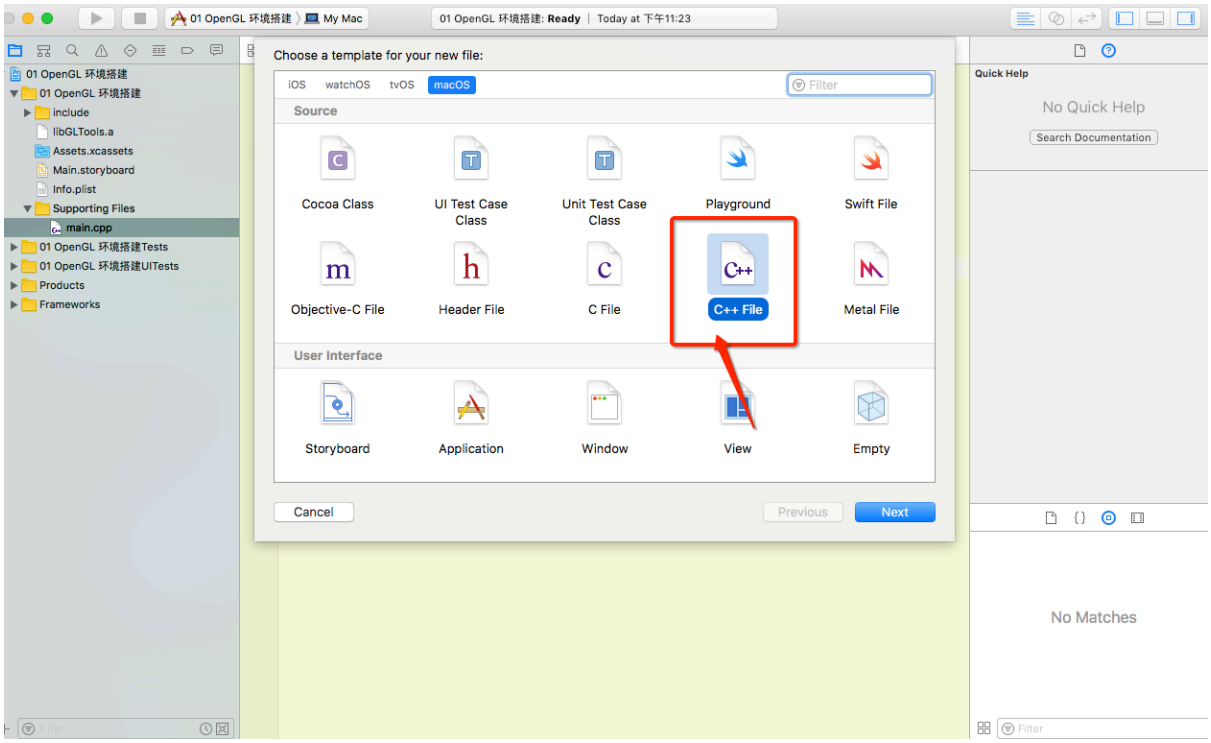
- 添加CLTools.h,glew.h
  在Bulid Settings 输入Header Search path 中拖入CLTool.h 和 glew.h 生成路径

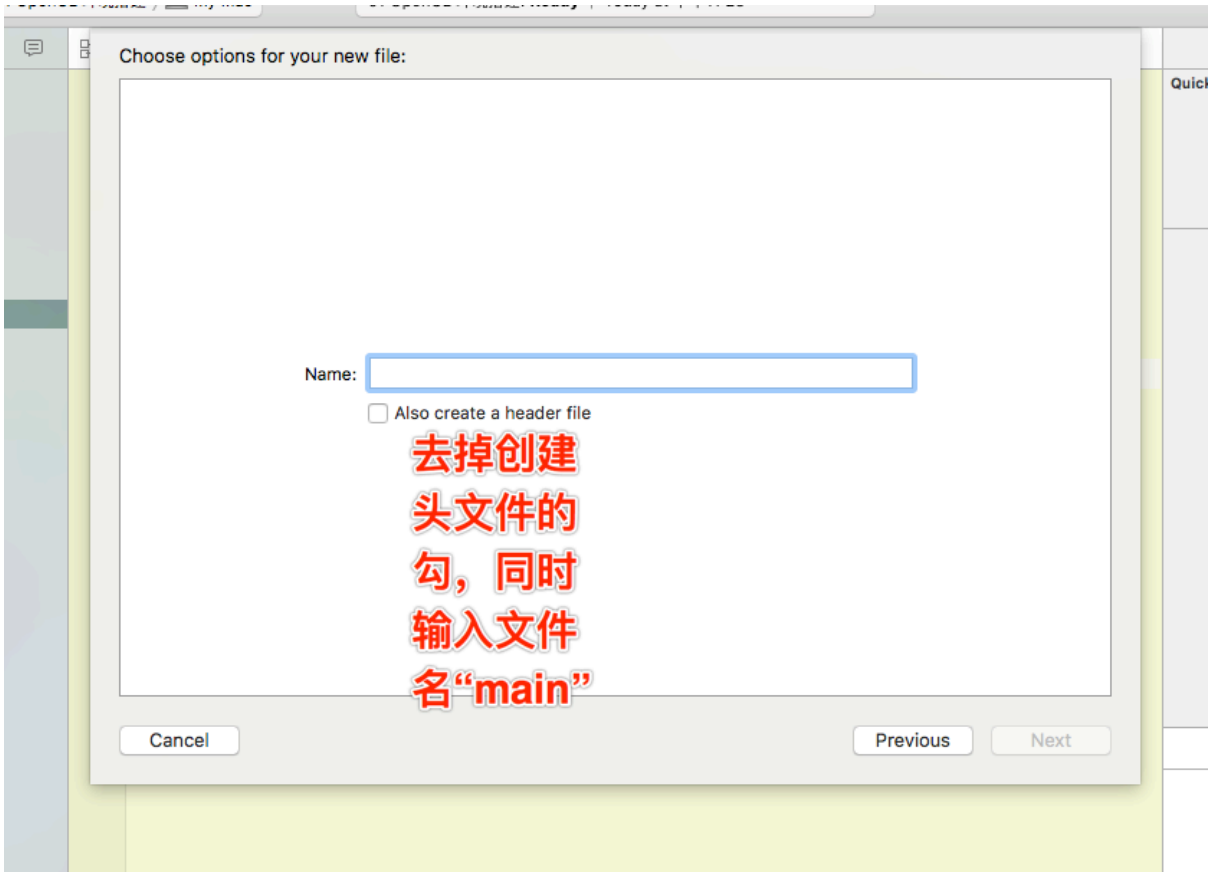


- libGLTools.a 直接拖到工程的Frameworks 文件里面，另外删除文件：AppDelegate.h 、 AppDelegate.m 、 main.m 、ViewController.h 、ViewController.m ; 创建 main.cpp文件


将include
文件夹拖
到项目中
将
libGLTool
s.a拖入到
Framewor
ks文件下
#include "GLShaderManager.h"
#include "GLTools.h"
#include <glut/glut.h>
GLBatch triangleBatch;
GLShaderManager shaderManager;
//窗口大小改变时接受新的宽度和高度，其中0,0代表窗口中视口的左下角坐标，w，h
ChangeSize(int w,int h)
glViewport(0,0, w, h);
}
//为程序作一次性的设置
SetupRC()
//设置背影颜色
glClearColor(0.0f,0.0f,1.0f,1.0f);
Symbol not found
Search Documentation
No Matches


//
// AppDelegate.h
// 01 OpenGL 环境搭建
//
// Created by CC老师 on 2017/8/28.
// Copyright © 2017年 Miss CC. All rights reserved.
//
#import <Cocoa/Cocoa.h>
@interface AppDelegate : NSObject <NSApplicationDelegate>
@end
删除选中
的文件
Declaration HEADER_SEARCH_PATHS
Value Type Path List
Description This is a list of paths to folders to be searched by the compiler for included or imported header files when compiling C, Objective-C, C++, or Objective-C++. Paths are delimited by whitespace, so any paths with spaces in them need to be properly quoted.
No Matches





在main.cpp中复制一下代码：

```
#include "GLTools.h"

#include <glut/glut.h>
```

```
GLBatch triangleBatch;

GLShaderManager shaderManager;

//窗口大小改变时接受新的宽度和高度，其中0,0代表窗口中视口的左下角坐标，w,h代表像素

void ChangeSize(int w,int h)

{

    glViewport(0,0, w, h);

}

//为程序作一次性的设置

void SetupRC()

{

    //设置背影颜色

    glClearColor(0.0f,0.0f,1.0f,1.0f);

    //初始化着色管理器

    shaderManager.InitializeStockShaders();

    //设置三角形，其中数组vVert包含所有3个顶点的x,y,笛卡尔坐标对。

    GLfloat vVerts[] = {

        -0.5f,0.0f,0.0f,
```

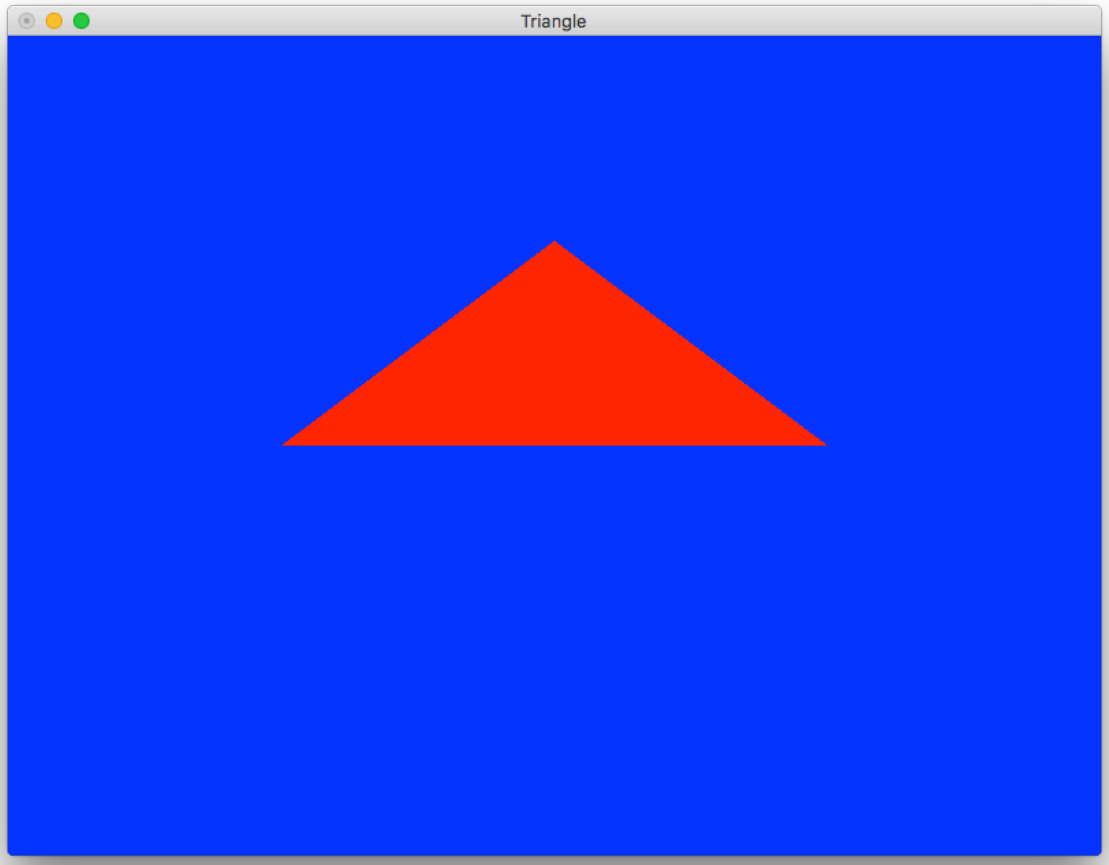
```
        0.5f,0.0f,0.0f,

        0.0f,0.5f,0.0f,

    };

    //批次处理

    triangleBatch.Begin(GL_TRIANGLES,3);

    triangleBatch.CopyVertexData3f(vVerts);

    triangleBatch.End();

}

//开始渲染

void RenderScene(void)

{

    //清除一个或一组特定的缓冲区

    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT|GL_STENCIL
_BUFFER_BIT);

    //设置一组浮点数来表示红色

    GLfloat vRed[] = {1.0f,0.0f,0.0f,1.0f};

    //传递到存储着色器，即GLT_SHADER_IDENTITY着色器，这个着色器只是使
用指定颜色以默认笛卡尔坐标第在屏幕上渲染几何图形

    shaderManager.UseStockShader(GLT_SHADER_IDENTITY,vRed);
```

```
    //提交着色器
    triangleBatch.Draw();
    //将在后台缓冲区进行渲染，然后在结束时交换到前台
    glutSwapBuffers();
}
int main(int argc,char* argv[])
{
    //设置当前工作目录，针对MAC OS X
    gltSetWorkingDirectory(argv[0]);
    //初始化GLUT库
    glutInit(&argc, argv);
    /*初始化双缓冲窗口，其中标志GLUT_DOUBLE、GLUT_RGBA、GLUT_DEPTH、GLUT_STENCIL分别指
     双缓冲窗口、RGBA颜色模式、深度测试、模板缓冲区*/
    glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGBA|GLUT_DEPTH|GLUT_STENCIL);
    //GLUT窗口大小，标题窗口
    glutInitWindowSize(800,600);
```

```
    glutCreateWindow("Triangle");

    //注册回调函数

    glutReshapeFunc(ChangeSize);

    glutDisplayFunc(RenderScene);

    //驱动程序的初始化中没有出现任何问题。

    GLenum err = glewInit();

    if(GLEW_OK != err) {

        fprintf(stderr,"glew error:%s\n",glewGetErrorString(er
r));

        return 1;

    }

    //调用SetupRC

    SetupRC();

    glutMainLoop();

    return 0;

}
```

- 编译，将文件<>系统引入，改为“”普通引入

效果图


Triangle