要实现3D引擎的跨平台功能. 必须研究OpenGLES的接口规范. 而OpenGLES 在[**iOS**](http://lib.csdn.net/base/ios)平台上的使用也有自身的特点.本文将详细比较两者在使用上的异同.

一. OpenGL ES对象及接口函数

    OpenGL ES中的对象分别用于处理两大类问题:   平台相关的问题和OpenGL渲染相关的问题.

    EGLDisplay与EGLSurface用于处理与平台相关的问题.

    EGLContext处理与OpenGL渲染相关的问题.

1. EGLDisplay可以简单理解为一个显示渲染结果的渲染窗口. EGLSurface可以理解为此窗口中的显示表面.只有画在此surface上的内容,才是渲染窗口中最后显示出来的画面.

1)   一般的应用平台在使用OpenGLES对象时, 只要使用以下API即可:

   eglGetDisplay,  eglInitialize,   eglChooseConfig,   eglCreateWindowSurface 等.

2)  但在Apple平台中, 对OpenGLES的使用略有不同.

   在IOS编程中必须要有一个UIWindow类实例对象,UIWindow就相当于上面的EGLDisplay.

   同时还必须要有一个从UIView类派生的类(可以称之为CustomView),UIView就相当于上面的EGLSurface. 并且必须将CustomView的成员变量layer, 由CALayer类型变为CAEAGLLayer类型. CAEAGLLayer成员的作用, 类似于eglChooseConfig函数. 用来设置目标surface的宽,高,像素格式及透明性等属性.设置完CAEAGLLayer后, 再以此对象的设置为参数, 调用renderbufferStorage来分配相应的内存空间(其作用类似于eglCreateWindowSurface?).

   需要明确的是, 一个UIView的最终显示结果, 是由它所包含的所有layer的画面叠加后形成的.

2. EGLContext

    与OpenGL渲染相关的内容由EGLContext对象来管理. 首先要调用eglCreateContext函数创建一个EGLContext对象,EGLContext内部采用双缓冲机制来实现渲染结果的显示.EGLContext有两个Framebuffer对象.一个做为back一个为front.当渲染完一帧后.通过调用eglSwapBuffers函数, EGLContext会自动交换两个Framebuffer的位置, 并且将front Framebuffer中的内容提交给相应的EGLSurface.  
在一个进程中,可以同时创建多个EGLContext对象(eglCreateContext),但当前context只能有一个, 需要通过eglMakeCurrent来设定.  
  
 在Apple平台中,使用情况稍有不同.  
    Apple平台不允许直接对Surface进行操作.这也就意味着在Apple中,不能通过调用eglSwapBuffers函数来直接实现渲染结果在目标surface上的更新.  
    在Apple平台中,首先要创建一个EAGLContext对象来替代EGLContext (不能通过eglCreateContext来生成), EAGLContext的作用与EGLContext是类似的.  
    然后,再创建相应的Framebuffer和Renderbuffer.

    Framebuffer象一个Renderbuffer集(它可以包含多个Renderbuffer对象).

    Renderbuffer有三种:  color Renderbuffer, depth Renderbuffer, stencil Renderbuffer.

    渲染结果是先输出到Framebuffer中,然后通过调用context的presentRenderbuffer,将Framebuffer上的内容提交给之前的CustumView.

二. [**ios**](http://lib.csdn.net/base/ios)中渲染一个3D场景的做法如下:

1. 生成一个OffScreen Framebuffer对象. 3D场景的渲染结果最终会由此Framebuffer给出.

    生成renderbuffer(或称color buffer), 并将其与Framebuffer挂接到一起.

　用glGenFramebuffers创建帧缓冲.

　用glGenRenderbuffers创建渲染缓冲.

　然后用glFramebufferRenderbuffer将Renderbuffer挂接到相应的Framebuffer上.

2. 生成一个CAEAGLLayer对象,

    作为图像最终输出窗口的UIView, 有一个成员变量CALayer, 要显示的图像都画在此对像上. 并且一个UIView可以有多个CALaye , 并且这些CALayer之间可以形成一个树形[**数据结构**](http://lib.csdn.net/base/datastructure), 所有layer中的结果数据叠加后形成最终的画面. 而不管这些 Core Animation Layer 是由 OpenGL ES 、Quartz 、还是其他什么图形库来描画的.

    如果要使输出画面只依赖OpenGLES渲染输出的图像,则要创建一个CAEAGLLayer对象. 并将不透明性设为YES .

3. 生成一个EAGLContext对象

     EAGLContext对象管理着渲染状态信息,命令和将被渲染的资源(An EAGLContext object manages the state information, commands, and resources needed to draw using OpenGL ES. ). 并且在将画面显示到context对象之前必须将一个framebuffer 绑定到此Context之上( To draw to an EAGL context, a complete framebuffer object must first be bound to the context. ) . 但找了许多资料也没看到如何将framebuffer与一个context绑定, 只看到将renderbuffer与context绑定的接口.

4.Framebuffer与CAEAGLLayer的关联

   在创建完以上三个对象后,

a. 先调用EAGLContext的类方法前之前生成的EAGLContext对象设定为当前context.

b. 然后调用EAGLContext对象的renderbufferStorage接口函数, 为之前生成的CAEAGLLayer对象分配空间.

   (　按IOS的文档的说法是:　Allocate its storage by calling the context’s [renderbufferStorage:fromDrawable:](http://developer.apple.com/library/ios/documentation/OpenGLES/Reference/EAGLContext_ClassRef/Reference/EAGLContext.html#//apple_ref/occ/instm/EAGLContext/renderbufferStorage:fromDrawable:) method, passing the layer object as the parameter.

　The width, height and pixel format are taken from the layer and used to allocate storage for the renderbuffer.　)

c. 在渲染之前先用glBindFramebuffer, 将之前用glGenFramebuffers创建的帧缓冲绑定为当前的Framebuffer(绑定到context上？).

    这样一来之后的所有的渲染命令就是针对当前绑定的Framebuffer(也就是针对新创建的Framebuffer)了.

    可以参见:http://miaoshuanghe.blog.163.com/blog/static/140130476201052453756439/ 中对Framebuffer的描述.

d. 在渲染完成之后，先调用

　glBindRenderbuffer(GL\_RENDERBUFFER, colorRenderbuffer);

　其作用是将Renderbuffer绑定到context上．（IOS的文档：binds the renderbuffer to the context）

　此时当前Framebuffer完全由renderbuffer控制（The color renderbuffer holds the completed frame, so all you need to do is present it to the user）．

    所谓hold的含义估计是这样来理解的：

　由于Framebuffer可以包括renderbuffer(或者称colorbuffer), depthbuffer, stencilbuffer.

　Framebuffer与renderbuffer/depthbuffer/stencilbuffer间的关系, 可以参见: http://xiaozu.renren.com/xiaozu/100134/335830019

　最终的输出结果(final output)由整个Framebuffer给出.

　所以，当renderbuffer "hold" Framebuffer时，Framebuffer的内容就是renderbuffer的内容．

e.  最后，调用EAGLContext的presentRenderbuffer．

　context presentRenderbuffer:GL\_RENDERBUFFER

　将最终渲染结果提交．

5. 需要说明的是OpenGLES中的

glGenBuffers

glBindBuffer

glBufferData

所针对的是资源数据缓冲.这类缓冲中存放的是作为渲染内容的外部资源数据, 比如顶点,法线等.