逻辑教育
Logic education

# Hello CC

OpenGL ES 主题 [1]

## 视觉班—OpenGL 过渡 OpenGL ES

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# OpenGL ES 简介

OpenGL ES (OpenGL for Embedded Systems) 是以手持和嵌入式为目标的高级3D图形应用程序编程接口(API). OpenGL ES 是目前智能手机中占据统治地位的图形API.支持的平台: iOS, Andriod , BlackBerry ,bada ,Linux ,Windows.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# OpenGL ES _ Apple

## OpenGL ES 苹果文档

The Open Graphics Library (OpenGL) is used for visualizing 2D and 3D data. It is a multipurpose open-standard graphics library that supports applications for 2D and 3D digital content creation, mechanical and architectural design, virtual prototyping, flight simulation, video games, and more. You use OpenGL to configure a 3D graphics pipeline and submit data to it. Vertices are transformed and lit, assembled into primitives, and rasterized to create a 2D image. OpenGL is designed to translate function calls into graphics commands that can be sent to underlying graphics hardware. Because this underlying hardware is dedicated to processing graphics commands, OpenGL drawing is typically very fast.

OpenGL for Embedded Systems (OpenGL ES) is a simplified version of OpenGL that eliminates redundant functionality to provide a library that is both easier to learn and easier to implement in mobile graphics hardware.

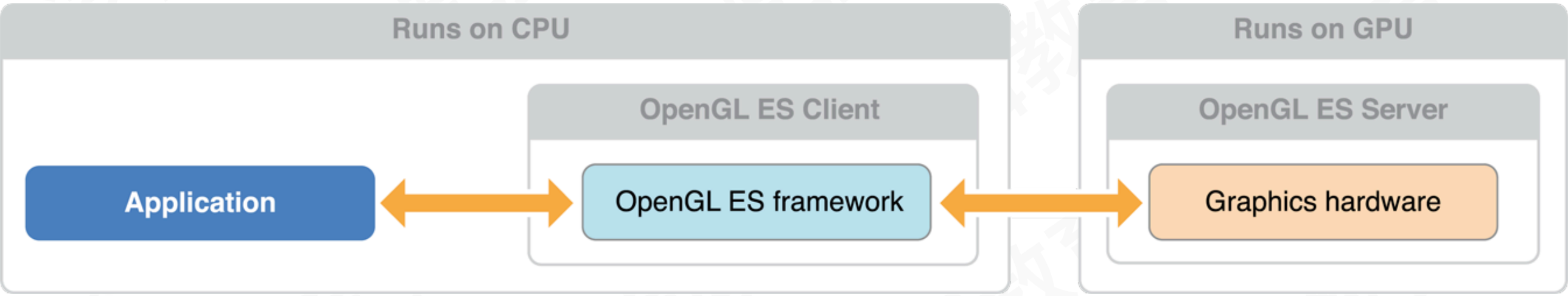
OpenGL ES 开放式图形库（OpenGL的）用于可视化的二维和三维数据。它是一个多功能开放标准图形库，支持2D和3D数字内容创建，机械和建筑设计，虚拟原型设计，飞行模拟，视频游戏等应用程序。您可以使用OpenGL配置3D图形管道并向其提交数据。顶点被变换和点亮，组合成图元，并光栅化以创建2D图像。OpenGL旨在将函数调用转换为可以发送到底层图形硬件的图形命令。由于此底层硬件专用于处理图形命令，因此OpenGL绘图通常非常快。

OpenGL for Embedded Systems（OpenGL ES）是OpenGL的简化版本，它消除了冗余功能，提供了一个既易于学习又更易于在移动图形硬件中实现的库。

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# OpenGL ES

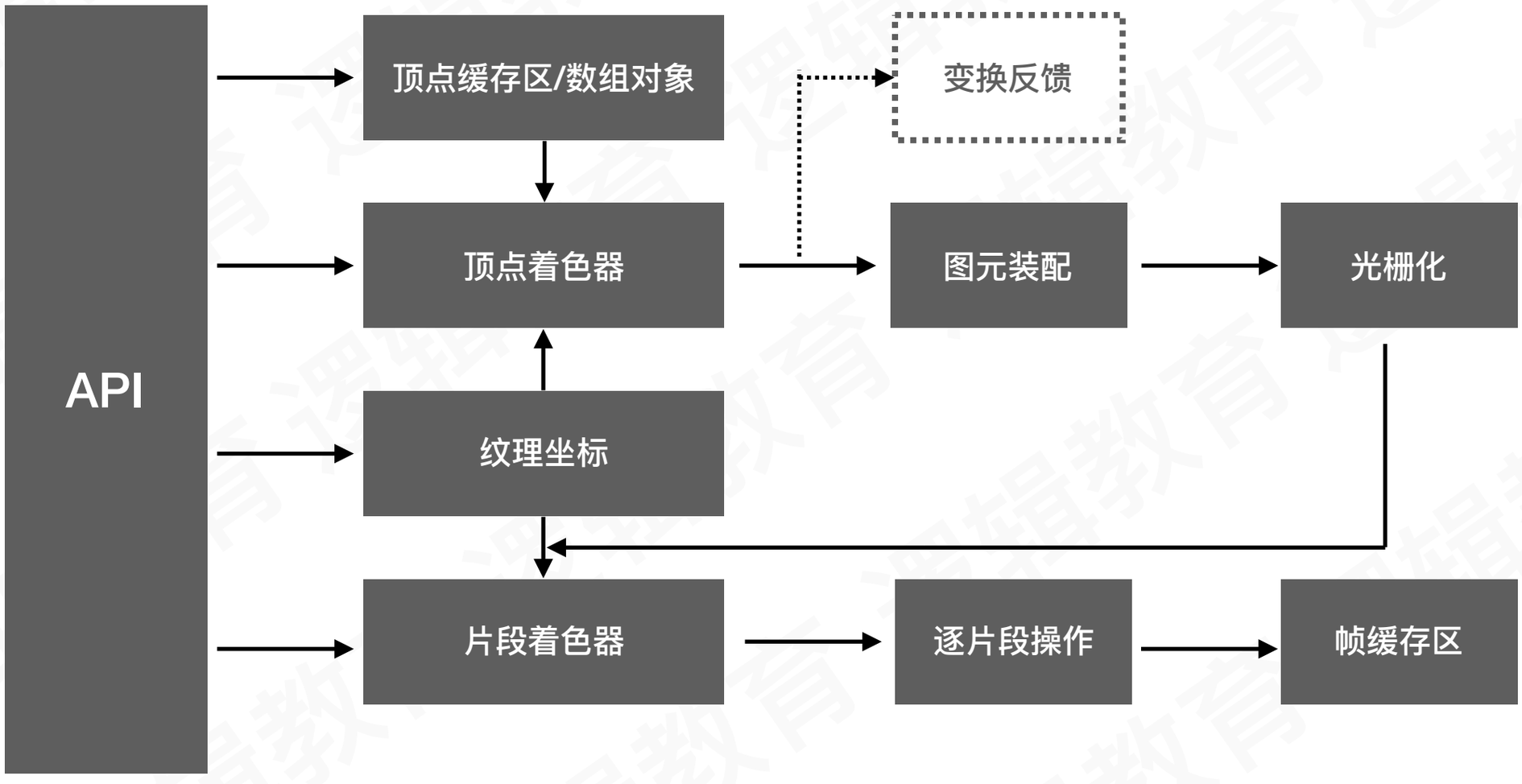



OpenGL ES允许应用程序利用底层图形处理器的强大功能。iOS设备上的GPU可以执行复杂的2D和3D绘图，以及最终图像中每个像素的复杂着色计算


课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

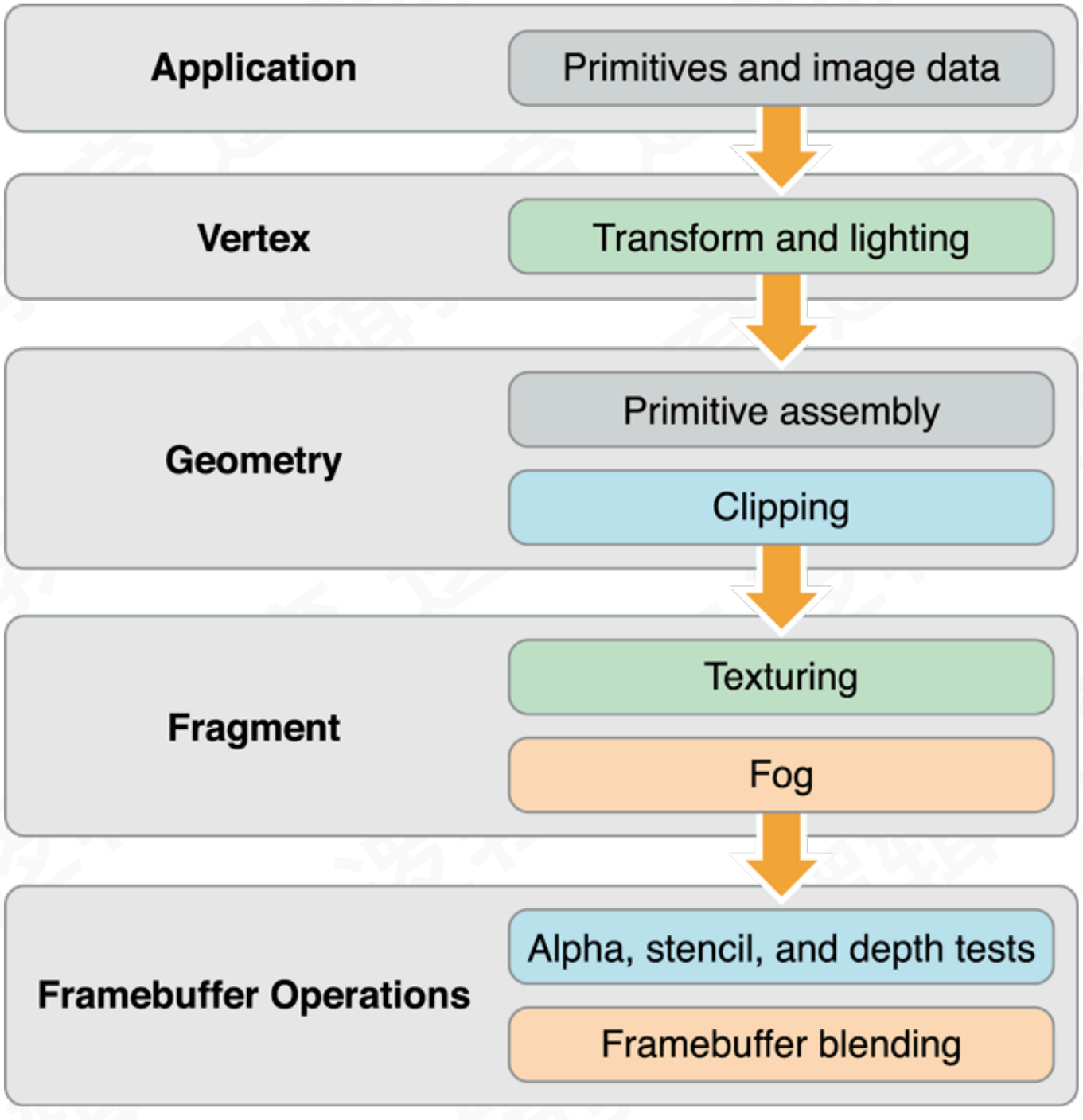
# OpenGL ES 3.0 图形管线




课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# OpenGL ES 图形管道




课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 顶点着色器

顶点着色器：

1. 着色器程序—描述顶点上执行操作的顶点着色器程序源代码/可执行文件

2.顶点着色器输入(属性) — 用顶点数组提供每个顶点的数据

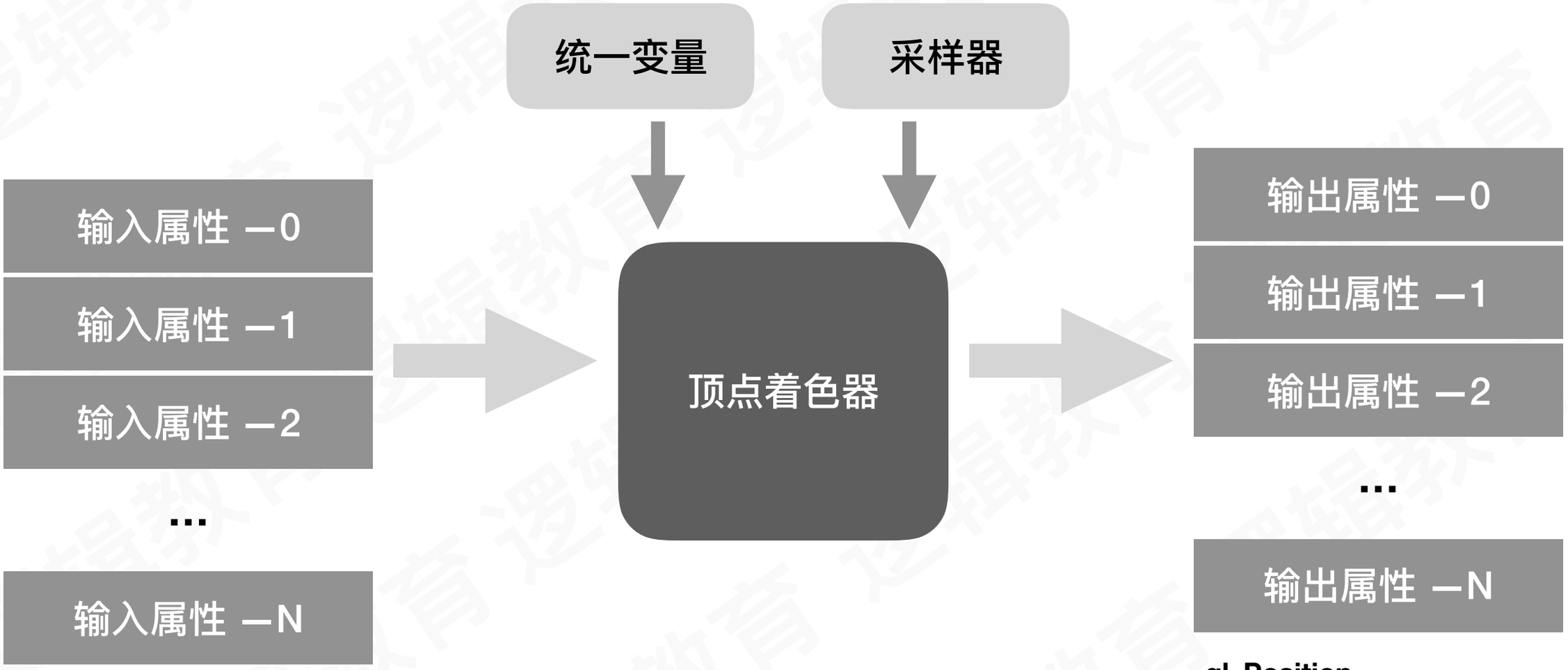
3.统一变量(uniform)—顶点/片元着色器使用的不变数据

4. 采样器—代表顶点着色器使用纹理的特殊统一变量类型.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# OpenGL ES 3.0 顶点着色器




课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 顶点着色业务:

顶点着色器 业务:

1. 矩阵变换位置

2.计算光照公式生成逐顶点颜色

3.生成/变换纹理坐标

总结: 它可以用于执行自定义计算,实施新的变换,照明或者传统的固定功能所不允许的基于顶点的效果.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 顶点着色代码案例:

```
attribute vec4 position;
attribute vec2 textCoordinate;
uniform mat4 rotateMatrix;
varying lowp vec2 varyTextCoord;
void main()
{
    varyTextCoord = textCoordinate;

    vec4 vPos = position;
    vPos = vPos * rotateMatrix;

    gl_Position = vPos;
}
```

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 图元装配

顶点着色器之后,下一个阶段就是图元装配.

图元(Primitive): 点,线,三角形等.

图元装配: 将顶点数据计算成一个个图元.在这个阶段会执行裁剪、透视分割和Viewport变换操作。

图元类型和顶点索确定将被渲染的单独图元。对于每个单独图元及其对应的顶点，图元装配阶段执行的操作包括：将顶点着色器的输出值执行裁剪、透视分割、视口变换后进入光栅化阶段。

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 光栅化

在这个阶段绘制对应的图元(点/线/三角形). 光栅化就是将图元转化成一组二维片段的过程.而这些转化的片段将由片元着色器处理.这些二维片段就是屏幕上可绘制的像素.


课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 片段着色器/片元着色器

片元着色器/片段着色器:

1. 着色器程序—描述片段上执行操作的片元着色器程序源代码/可执行文件

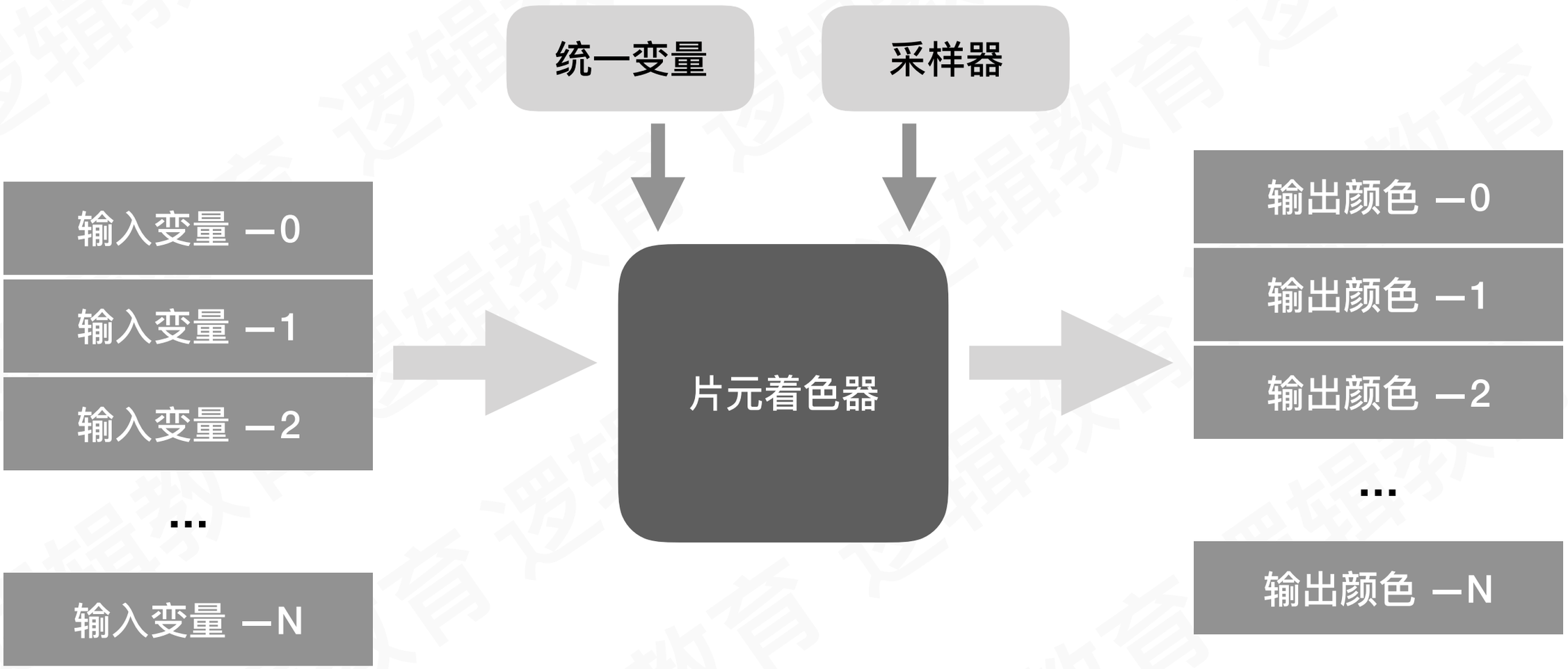
2.输入变量—光栅化单元用插值为每个片段生成的顶点着色器输出

3.统一变量(uniform)—顶点/片元着色器使用的不变数据

4. 采样器—代表片元着色器使用纹理的特殊统一变量类型.

课程研发:CC老师
课程授课:CC老师

# OpenGL ES 3.0 片段着色器/片元着色器




**gl_FragColor**

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 片元着色器业务:

片元着色器 业务:

1. 计算颜色

2. 获取纹理值

3. 往像素点中填充颜色值(纹理值/颜色值);

总结: 它可以用于图片/视频/图形中每个像素的颜色填充(比如给视频添加滤镜,实际上就是将视频中每个图片的像素点颜色填充进行修改.)

课程研发:CC老师
课程授课:CC老师

# 片元着色代码案例:

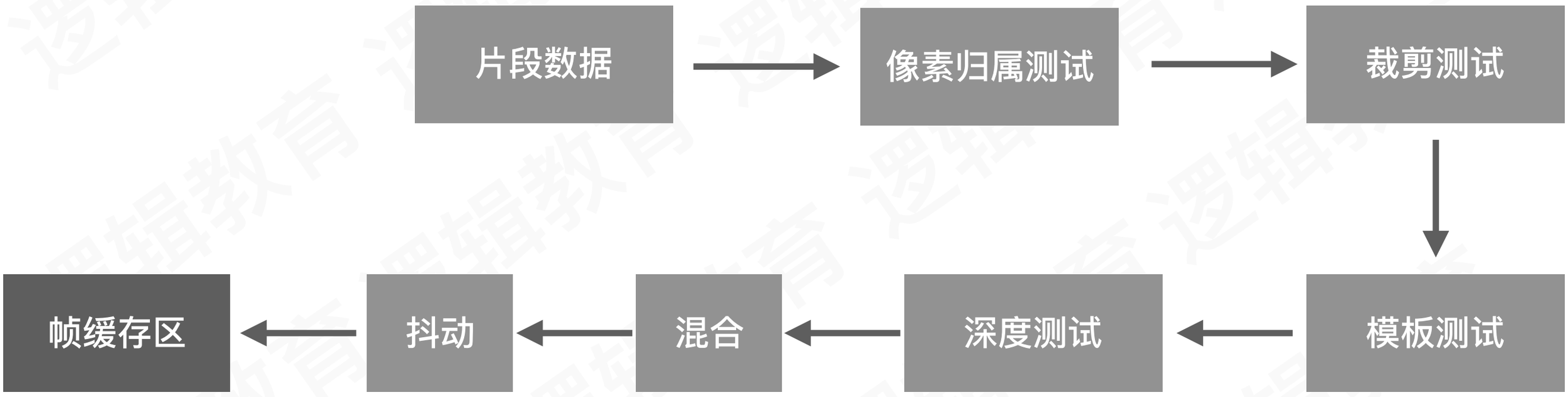
```
varying lowp vec2 varyTextCoord;
uniform sampler2D colorMap;

void main()
{
    gl_FragColor = texture2D(colorMap, varyTextCoord);
}
```

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 逐片段操作




课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 逐片段操作

- 像素归属测试: 确定帧缓存区中位置($X_W$,$Y_W$)的像素目前是不是归属于OpenGL ES所有.例如,如果一个显示OpenGL ES帧缓存区View被另外一个View 所遮蔽.则窗口系统可以确定被遮蔽的像素不属于OpenGL ES 上下文.从而不全显示这些像素. 而像素归属测试是OpenGL ES 的一部分,它不由开发者开人为控制,而是由OpenGL ES 内部进行.
- 裁剪测试: 裁剪测试确定($X_W$,$Y_W$)是否位于作为OpenGL ES状态的一部分裁剪矩形范围内.如果该片段位于裁剪区域之外,则被抛弃.
- 深度测试: 输入片段的深度值进步比较,确定片段是否拒绝测试
- 混合: 混合将新生成的片段颜色与保存在帧缓存的位置的颜色值组合起来.
- 抖动: 抖动可用于最小化因为使用有限精度在帧缓存区中保存颜色值而产生的伪像.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# EGL (Embedded Graphics Library )

- OpenGL ES 命令需要渲染上下文和绘制表面才能完成图形图像的绘制.
- 渲染上下文: 存储相关OpenGL ES 状态.
- 绘制表面: 是用于绘制图元的表面,它指定渲染所需要的缓存区类型,例如颜色缓存区,深度缓冲区和模板缓存区.
- OpenGL ES API 并没有提供如何创建渲染上下文或者上下文如何连接到原生窗口系统. EGL 是Khronos 渲染API(如OpenGL ES) 和原生窗口系统之间的接口. 唯一支持OpenGL ES 却不支持EGL 的平台是iOS. Apple 提供自己的EGL API的iOS实现,称为EAGL.
- 因为每个窗口系统都有不同的定义,所以EGL提供基本的不透明类型—EGLDisplay, 这个类型封装了所有系统相关性,用于和原生窗口系统接口.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

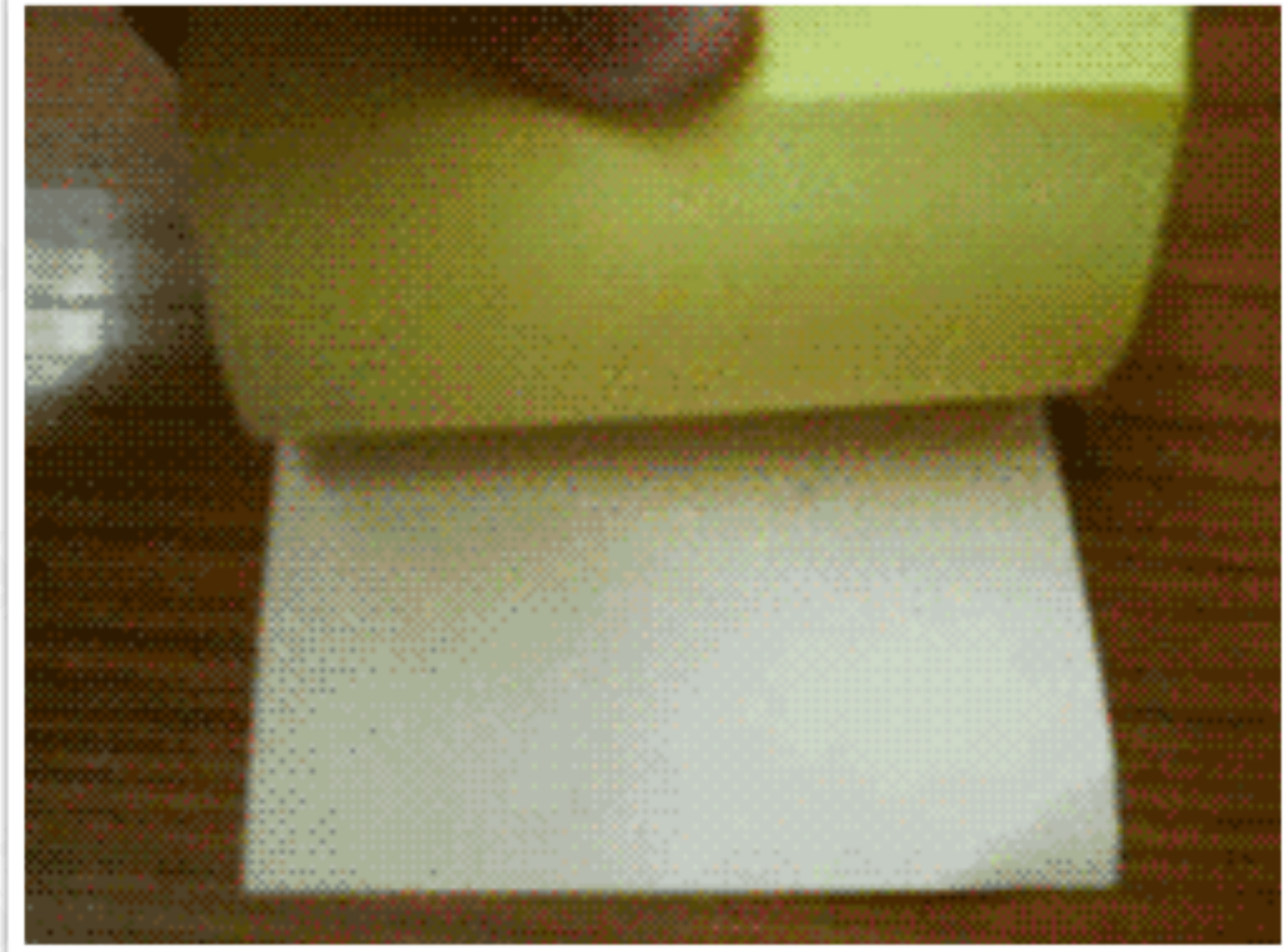
# EGL (Embedded Graphics Library )

由于OpenGL ES是基于C的API，因此它非常便携且受到广泛支持。作为C API，它与Objective-C Cocoa Touch应用程序无缝集成。OpenGL ES规范没有定义窗口层; 相反，托管操作系统必须提供函数来创建一个接受命令的OpenGL ES 渲染上下文和一个帧缓冲区，其中写入任何绘图命令的结果。在iOS上使用OpenGL ES需要使用iOS类来设置和呈现绘图表面，并使用平台中立的API来呈现其内容。

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

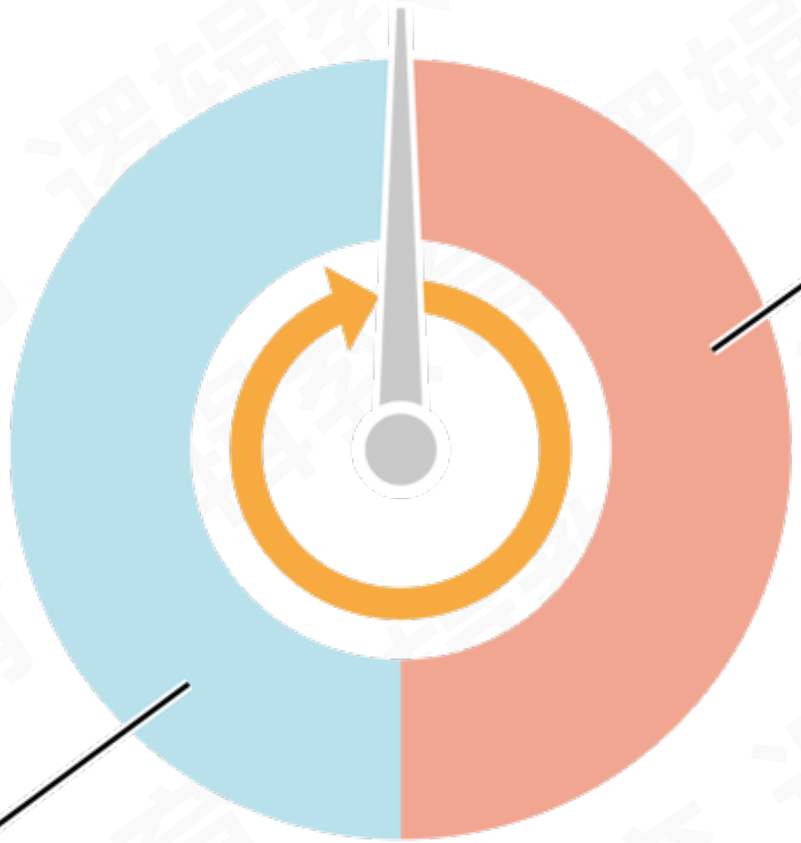
# 计算机对于动画的实现



课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 理解动画循环



**Update**

```
angle +=
    rate * timeSinceLastUpdate;
matrix =
    GLKMatrix4MakeRotation(
     angle, 0, 1, 0);
```

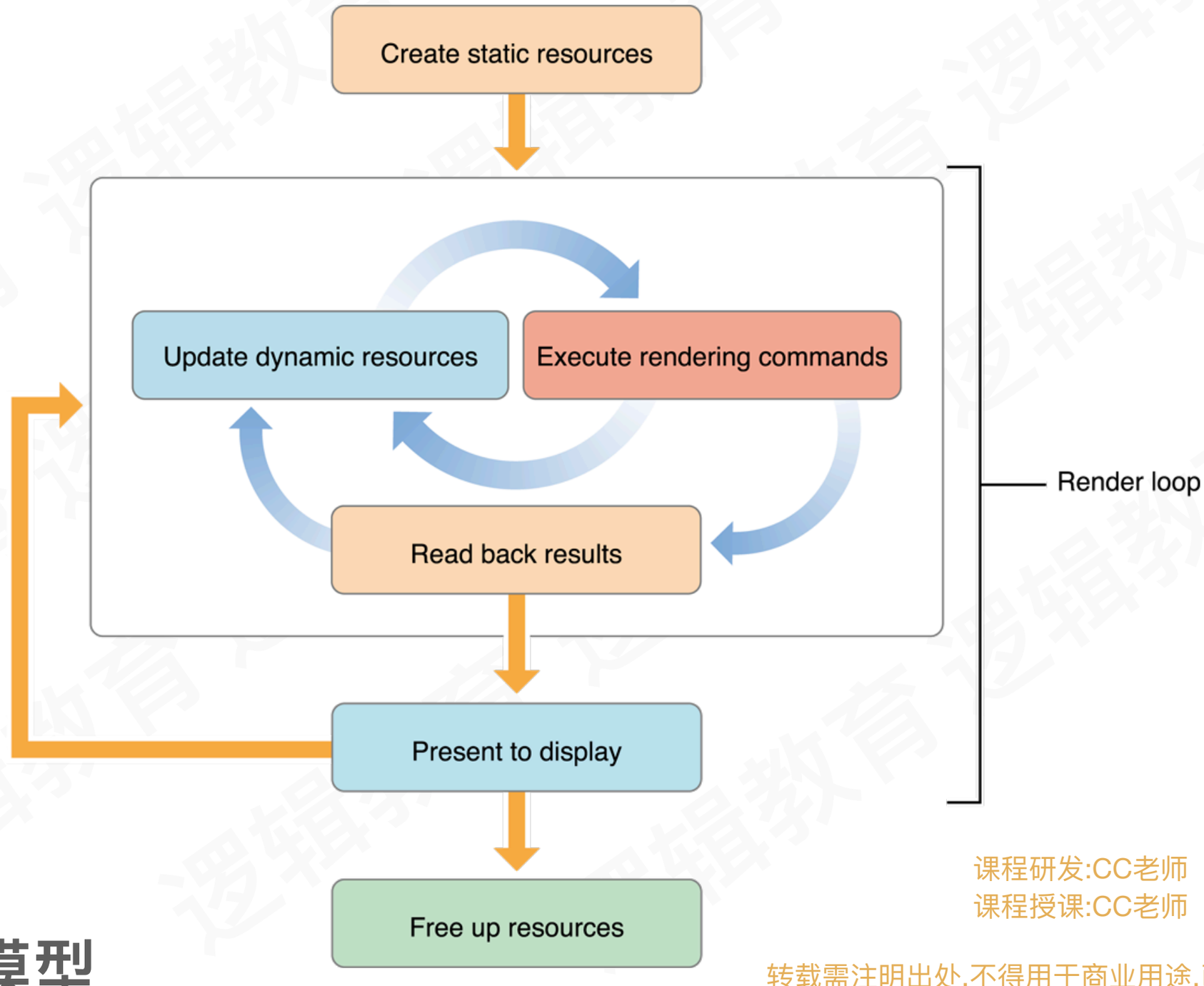
**Display**

```
glUniformMatrix4fv(
    uMVP, 1, 0, matrix.m);
glDrawElements(...);
```

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# OpenGL ES 显示器执行动画的应用程序流程




课程研发:CC老师
课程授课:CC老师

## 管理资源APP模型

逻辑教育
Logic education

# GLKit 框架概述

GLKit 框架的设计目标是为了简化基于OpenGL / OpenGL ES 的应用开发. 。它的出现加快OpenGL ES或OpenGL应用程序开发。 使用数学库，背景纹理加载，预先创建的着色器效果，以及标准视图和视图控制器来实现渲染循环。

GLKit框架提供了功能和类，可以减少创建新的基于着色器的应用程序所需的工作量，或者支持依赖早期版本的OpenGL ES或OpenGL提供的固定函数顶点或片段处理的现有应用程序。

GLKView 提供绘制场所(View)

GLKViewController(扩展于标准的UIKit 设计模式. 用于绘制视图内容的管理与呈现.)

苹果弃用OpenGL ES ,但iOS开发者可以继续使用.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 苹果文档介绍

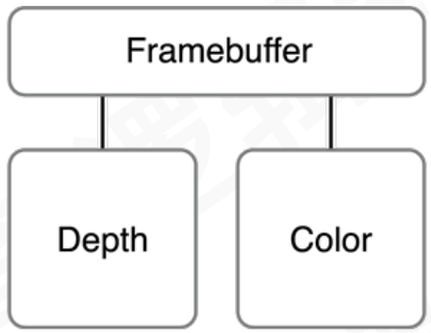
The `GLKView` class provides an OpenGL ES–based equivalent of the standard `UIView` drawing cycle. A `UIView` instance automatically configures its graphics context so that your `drawRect:` implementation need only perform Quartz 2D drawing commands, and a `GLKView` instance automatically configures itself so that your drawing method need only perform OpenGL ES drawing commands. The `GLKView` class provides this functionality by maintaining a framebuffer object that holds the results of your OpenGL ES drawing commands, and then automatically presents them to Core Animation once your drawing method returns.

Like a standard UIKit view, a GLKit view renders its content on demand. When your view is first displayed, it calls your drawing method—Core Animation caches the rendered output and displays it whenever your view is shown. When you want to change the contents of your view, call its `setNeedsDisplay`method and the view again calls your drawing method, caches the resulting image, and presents it on screen. This approach is useful when the data used to render an image changes infrequently or only in response to user action. By rendering new view contents only when you need to, you conserve battery power on the device and leave more time for the device to perform other actions.

课程研发:CC老师
课程授课:CC老师

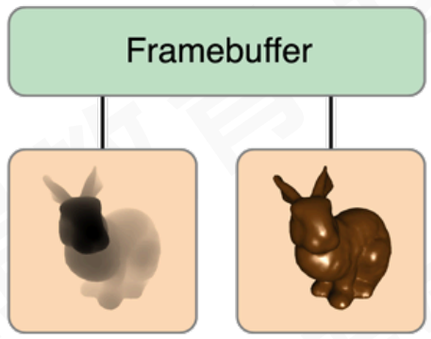
# 使用GLKit视图呈现OpenGL ES内容

1. GLKView prepares
   OpenGL ES infrastructure

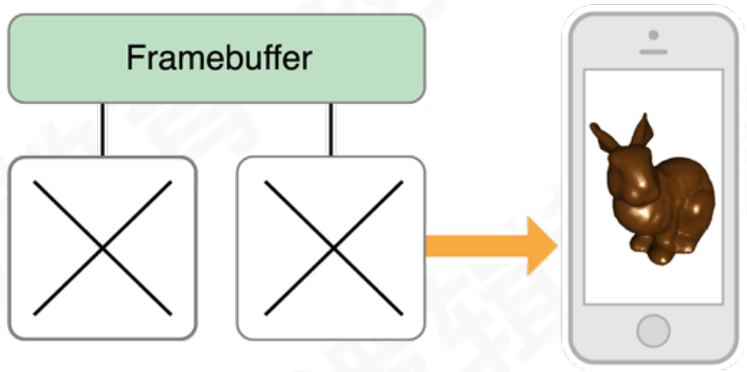



2. Draw into the framebuffer

```
{ // your code
  glClear(...)
  glUniform...(...)
  glBind...(...)
  glDraw...(...)
}
```




3. GLKView presents the image




课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 配置GLKit视图(代码清单9-1)

```
- (void) viewDidLoad{

    [super viewDidLoad];

    //创建OpenGL ES上下文并将其分配给从故事板加载的视图
    GLKView * view = (GLKView *) self.view;
    view.context = [[EAGLContext alloc] initWithAPI: kEAGLRenderingAPIOpenGLES2];

    //配置视图创建的渲染缓冲区
    view.drawableColorFormat = GLKViewDrawableColorFormatRGBA8888;
    view.drawableDepthFormat = GLKViewDrawableDepthFormat24;
    view.drawableStencilFormat = GLKViewDrawableStencilFormat8;

    //启用多重采样
    view.drawableMultisample = GLKViewDrawableMultisample4X;

}
```

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# 配置GLKit视图(代码清单9-2)

```
-(void)drawRect:(CGRect)rect
{
    //清除帧缓冲区
    glClearColor (0.0f, 0.0f, 0.1f, 1.0f) ;
    glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT) ;

   //使用先前配置的纹理，着色器和顶点数组绘制
    glBindTexture (GL_TEXTURE_2D, _planetTexture) ;
    glUseProgram (_diffuseShading) ;
    glUniformMatrix4fv (_uniformModelViewProjectionMatrix, 1,0,
_modelViewProjectionMatrix.m) ;
    glBindVertexArrayOES (_planetMesh) ;
    glDrawElements (GL_TRIANGLE_STRIP, 256, GL_UNSIGNED_SHORT) ;

}
```

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 功能

- 加载纹理
- 提供高性能的数学运算
- 提供常见的着色器
- 提供视图以及视图控制器.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 纹理加载

## GLKTextureInfo 创建OpenGL 纹理信息.

- name : OpenGL 上下文中纹理名称
- target : 纹理绑定的目标
- height : 加载的纹理高度
- width : 加载纹理的宽度
- textureOrigin : 加载纹理中的原点位置
- alphaState: 加载纹理中alpha分量状态
- containsMipmaps: 布尔值,加载的纹理是否包含mip贴图

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 纹理加载

GLTextureLoader 简化从各种资源文件中加载纹理.

- 初始化
  - - initWithSharegroup: 初始化一个新的纹理加载到对象中
  - - initWithShareContext: 初始化一个新的纹理加载对象
- 从文件中加载处理
  - + textureWithContentsOfFile:options:errer: 从文件加载2D纹理图像并从数据中创建新的纹理
  - - textureWithContentsOfFile:options:queue:completionHandler: 从文件中异步加载2D纹理图像,并根据数据创建新纹理

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 纹理加载

GLTextureLoader 简化从各种资源文件中加载纹理.

- 从URL加载纹理
  - - textureWithContentsOfURL:options:error: 从URL 加载2D纹理图像并从数据创建新纹理
  - - textureWithContentsOfURL:options:queue:completionHandler: 从URL异步加载2D纹理图像,并根据数据创建新纹理.
- 从内存中表示创建纹理
  - + textureWithContentsOfData:options:errer: 从内存空间加载2D纹理图像,并根据数据创建新纹理
  - - textureWithContentsOfData:options:queue:completionHandler:从内存空间异步加载2D纹理图像,并从数据中创建新纹理

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 纹理加载

GLTextureLoader 简化从各种资源文件中加载纹理.

- 从CGImages创建纹理
  - - textureWithCGImage:options:error: 从Quartz图像 加载2D纹理图像并从数据创建新纹理
  - - textureWithCGImage:options:queue:completionHandler: 从Quartz图像异步加载2D纹理图像,并根据数据创建新纹理.
- 从URL加载多维创建纹理
  - + cabeMapWithContentsOfURL:options:errer: 从单个URL加载立方体贴图纹理图像,并根据数据创建新纹理
  - - cabeMapWithContentsOfURL:options:queue:completionHandler:从单个URL异步加载立方体贴图纹理图像,并根据数据创建新纹理

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 纹理加载

GLTextureLoader 简化从各种资源文件中加载纹理.

- 从文件加载多维数据创建纹理
  - + cubeMapWithContentsOfFile:options:errer: 从单个文件加载立方体贴图纹理对象,并从数据中创建新纹理
  - - cubeMapWithContentsOfFile:options:queue:completionHandler:从单个文件异步加载立方体贴图纹理对象,并从数据中创建新纹理
  - + cubeMapWithContentsOfFiles:options:errer: 从一系列文件中加载立方体贴图纹理图像,并从数据总创建新纹理
  - - cubeMapWithContentsOfFiles:options:options:queue:completionHandler: 从一系列文件异步加载立方体贴图纹理图像,并从数据中创建新纹理

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKView 使用OpenGL ES 绘制内容的视图默认实现

- 初始化视图
  - - initWithFrame:context: 初始化新视图
- 代理
  - delegate 视图的代理
- 配置帧缓存区对象
  - drawableColorFormat  颜色渲染缓存区格式
  - drawableDepthFormat 深度渲染缓存区格式
  - drawableStencilFormat 模板渲染缓存区的格式
  - drawableMultisample  多重采样缓存区的格式
- 帧缓存区属性
  - drawableHeight 底层缓存区对象的高度(以像素为单位)
  - drawableWidth 底层缓存区对象的宽度(以像素为单位)

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKView 使用OpenGL ES 绘制内容的视图默认实现

- 绘制视图的内容
  - context  绘制视图内容时使用的OpenGL ES 上下文
  - - bindDrawable 将底层FrameBuffer 对象绑定到OpenGL ES
  - enableSetNeedsDisplay 布尔值,指定视图是否响应使得视图内容无效的消息
  - - display  立即重绘视图内容
  - snapshot 绘制视图内容并将其作为新图像对象返回
- 删除视图FrameBuffer对象
  - - deleteDrawable 删除与视图关联的可绘制对象

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKViewDelegate 用于GLKView 对象回调方法

- 绘制视图的内容
  - - glkView:drawInRect:  绘制视图内容 (必须实现代理)

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKViewController 管理OpenGL ES 渲染循环的视图控制器

- 更新
  - - (void) update 更新视图内容
  - - (void) glkViewControllerUpdate:
- 配置帧速率
  - preferredFramesPerSecond 视图控制器调用视图以及更新视图内容的速率
  - framesPerSencond 视图控制器调用视图以及更新其内容的实际速率
- 配置GLKViewController 代理
  - delegate 视图控制器的代理

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKViewController 管理OpenGL ES 渲染循环的视图控制器

- 控制帧更新
  - paused 布尔值,渲染循环是否已暂停
  - pausedOnWillResignActive 布尔值,当前程序重新激活活动状态时视图控制器是否自动暂停渲染循环
  - resumeOnDidBecomeActive 布尔值,当前程序变为活动状态时视图控制是否自动恢复呈现循环

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKViewController 管理OpenGL ES 渲染循环的视图控制器

- 获取有关View 更新信息
  - frameDisplayed 视图控制器自创建以来发送的帧更新数
  - timeSinceFirstResume 自视图控制器第一次恢复发送更新事件以来经过的时间量
  - timeSinceLastResume 自上次视图控制器恢复发送更新事件以来更新的时间量
  - timeSinceLastUpdate 自上次视图控制器调用委托方法以及经过的时间量 glkViewControllerUpdate:
  - timeSinceLastDraw 自上次视图控制器调用视图display 方法以来经过的时间量.

课程研发:CC老师
课程授课:CC老师

# GLKit OpenGL ES 视图渲染

## GLKViewControllerDelegate 渲染循环回调方法

- 处理更新事件
  - - glkViewControllerUpdate: 在显示每个帧之前调用

- 暂停/恢复通知
  - - glkViewController:willPause: 在渲染循环暂停或恢复之前调用.

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKBaseEffect 一种简单光照/着色系统,用于基于着色器OpenGL 渲染

- 命名Effect
  - label 给Effect(效果)命名

- 配置模型视图转换
  - transform 绑定效果时应用于顶点数据的模型视图,投影和纹理变换

- 配置光照效果
  - lightingType 用于计算每个片段的光照策略,GLKLightingType
  - GLKLightingType
    - GLKLightingTypePerVertex 表示在三角形中每个顶点执行光照计算,然后在三角形进行插值
    - GLKLightingTypePerPixel 表示光照计算的输入在三角形内插入,并且在每个片段执行光照计算

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKBaseEffect 一种简单光照/着色系统,用于基于着色器OpenGL 渲染

- 配置光照
- lightModelTwoSided 布尔值,表示为基元的两侧计算光照
- material 计算渲染图元光照使用的材质属性
- lightModelAmbientColor 环境颜色,应用效果渲染的所有图元.
- light0 场景中第一个光照属性
- light1 场景中第二个光照属性
- light2 场景中第三个光照属性

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKBaseEffect 一种简单光照/着色系统,用于基于着色器OpenGL 渲染

- 配置纹理
  - texture2d0 第一个纹理属性
  - texture2d1 第二个纹理属性
  - textureOrder 纹理应用于渲染图元的顺序
- 配置雾化
  - fog 应用于场景的雾属性
- 配置颜色信息
  - colorMaterialEnable 布尔值,表示计算光照与材质交互时是否使用颜色顶点属性
  - useConstantColor 布尔值,指示是否使用常量颜色
  - constantColor 不提供每个顶点颜色数据时使用常量颜色

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit OpenGL ES 视图渲染

## GLKBaseEffect 一种简单光照/着色系统,用于基于着色器OpenGL 渲染

- 准备绘制效果
  - - prepareToDraw 准备渲染效果

课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 案例实现(01)

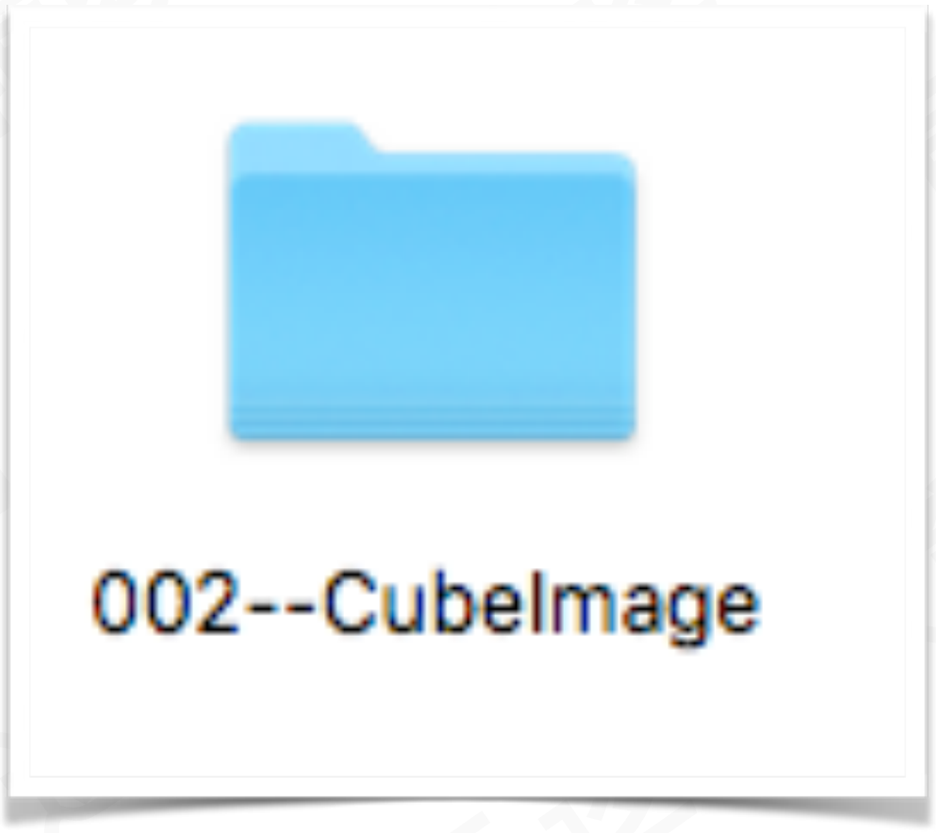
## 使用GLKit 在屏幕上加载一张图片


课程研发:CC老师
课程授课:CC老师

逻辑教育
Logic education

# GLKit 案例实现(02)

## 使用GLKit 在屏幕上加载正方体




课程研发:CC老师
课程授课:CC老师