MyArrow的专栏

个人资料

Arrow

访问： 1246631次
积分： 14634
等级： BLOG 7
排名： 第298名

原创： 224篇 转载： 173篇
译文： 4篇 评论： 261条

OpenGL ES2.0 基本编程

分类： OpenGL ES 2012-06-28 11:52 4543人阅读 评论(1) 收藏 举报

编程 buffer command server primitive

1. EGL

OpenGL ES命令需要一个rendering context和一个drawing surface。

Rendering Context: 保存当前的OpenGL ES状态。

Drawing Surface: 是原语(primitive)画图的Surface。它指定了渲染的buffer类型，如：color buffer，depth buffer和stencil buffer；同时它也指定了每个需要的buffer的位深度(bit depth)。

EGL是OpenGL ES API与Native Window System之间的接口。在OpenGL ES执行render之前，需要EGL做以下工作：

• 查询设备上可得到的显示设备，并初始化它们。

• 创建一个Rendering Surface(渲染表面)。EGL可以创建屏幕上的表面(on-srceen surface)或离线屏幕表面off-screen surface，屏幕上的表面连接到本地窗口系统；而离线屏幕表面不显示，但可以用于渲染表面(rendering surface)的像素缓冲区。

• 创建一个rendering context(渲染环境)。在真正开始画图之前，需要把渲染环境连接到渲染表面。

1.1 EGL 数据类型

Data Type	C-Language Type	EGL Type
32-bit integer	int	EGLint
32-bit unsigned integer	unsigned int	EGLBoolean, EGLenum
32-bit pointer	void *	EGLConfig, EGLContext, EGLDisplay, EGLSurface, EGLClientBuffer

2. OpengGL ES命令后缀和参数数据类型

Data Type Suffix	Data Type	C-Language Type	GL Type
b	8-bit signed integer	signed char	GLbyte
ub	8-bit unsigned integer	unsigned char	GLubyte, GLboolean
s	16-bit signed integer	short	GLshort
us	16-bit unsigned integer	unsigned short	GLushort
i	32-bit signed integer	int	GLint

ALSA (13)
V4L2&USB (14)
内存管理 (4)
Linux API (2)
Linux安全 (9)
云计算 (1)
异构计算 (1)
cocos2d-x (17)
Unity3D (18)
H.265/HEVC (4)
3DS Max (3)
Photoshop (1)

文章存档

2015年05月 (3)
2015年04月 (2)
2015年03月 (5)
2015年02月 (4)
2015年01月 (10)
展开

阅读排行

WiFi基本知识 (37159)
Android 4.0 事件输入(Ev (31420)
Android4.0.3 显示系统深 (24101)
Linux pipe函数 (22088)
Android4.x 如何处理Pow (19194)
Android WiFi--系统架构 (15798)
Linux inotify功能及实现原 (14156)
USB枚举过程 (13987)
波长与频率的关系 (12320)
android surfaceflinger研究 (10899)

评论排行

Android4.0.3 显示系统深 (25)
Android 4.0 事件输入(Ev (21)
WiFi基本知识 (16)
Android4.x 如何处理Pow (12)
USB协通讯议--深入理解 (10)
USB枚举过程 (8)
Linux内存管理--基本概念 (7)
Linux Wireless架构总结 (7)
Mali GPU OpenGL ES 总 (7)
Android WiFi--系统架构 (7)

推荐文章

ui	32-bit unsigned integer	unsigned int	GLuint, GLbitfield, GLenum
x	16.16 fixed point	int	GLfixed
f	32-bit floating point	float	GLfloat, GLclampf

GLvoid是OpenGL ES命令可接受的指针。

2.1 OpenGL ES基本错误码

错误码可通过GLenum glGetError(void)函数获取，如果当前错误码的值不为GL_NO_ERROR，则新产生的错误码不能被保存。

Error Code	Description
GL_NO_ERROR	No error has been generated since the last call to glGetError.
GL_INVALID_ENUM	A GLenum argument is out of range. The command that generated the error is ignored.
GL_INVALID_VALUE	A numeric argument is out of range. The command that generated the error is ignored.
GL_INVALID_OPERATION	The specific command cannot be performed in the current OpenGL ES state. The command that generated the error is ignored.
GL_OUT_OF_MEMORY	There is insufficient memory to execute this command. The state of the OpenGL ES pipeline is considered to be undefined if this error is encountered except for the current error code.

3. Flush和Finish

OpenGL ES2.0 API继承了OpenGL的C-S（客户端-服务器）模式。应用程序（客户端）发布命令，则Server负责执行处理。且不是应用程序每发一个命令都被及时地发送给Server。在设备中CPU负责运行游戏的逻辑，并向GPU（硬件显卡或是软件模拟的显卡）发送绘图指令。在这种架构下，CPU和GPU分别充当客户端与服务器端的角色。

glFlush命令把当前OpenGL ES环境中的命令进行刷新，然后发送给Server。glFlush只是把命令发送给Server，但并不等待执行完成。如果需要等到Server执行完成时才返回，则需要调用glFinish，但它严重影响性能。

eglSwapBuffers中调用了glFlush。

4. 基本的状态管理

管道的每个阶段都有自己的一些状态，且每个状态有对应的值，这些状态值可以通过以下两个函数进行修改：

void glEnable(GLenum cap)
void glDisable(GLenum cap)

在初始状态时，除GL_DITHER（初始值为GL_TRUE）之外，其它每个状态的初始值都为GL_FALSE。这些状态值被保存在EGLcontext中。其状态值可通过glIsEnabled（GLboolean glIsEnabled(GLenum cap)）来进行查询。