RGU Game Engine Design Reference

2023 Admenri. The RGU Authors.

## 概述

**核心组件：**

窗口及操作系统访问服务（SDL）

图像渲染部分（OpenGL ES 2.0）

脚本执行部分（Binding）

音频服务（SDL Mixer）

**辅助组件：**

视频解码（VPX VP8/9）

文字软渲染（SDL TTF）

图像编解码（SDL Image）

## 源代码组成

base – 定义SDL库上层封装，操作系统特定API跨平台封装，线程与任务实现。

gpu – 定义具体渲染器的实现代码

renderer – 渲染过程及渲染对象管理

worker – 多线程架构调度及各组件调度管理

media – 音视频解码相关代码

build – 构建工具链相关工具源码

third\_party – 第三方库调用链接方式或源码

app – 应用程序入口

## 线程及任务机制

任务类中均含有特定被执行的任务（Task），通过base::BindOnce进行闭包创建绑定以投递到目标worker线程运行任务。

## 图像渲染

规定每个活动块均为Drawable的子类，父类DrawableContainer定义对象的z轴概念及具体实现（更改z轴，遍历实现容器内的绘制操作）。

规定Viewport类继承DrawableContainer与Drawable，Graphics中存储根Drawable类为RootDrawable。

规定Bitmap类使用FrameBufferObject技术实现图像绘制，需定义一个Texture和FrameBuffer对象。

规定全部Drawable经过Graphics::update后的绘制目标为ScreenFrameBuffer，通过FBOBlt等转移到前端窗口画面，此前经过明暗度等处理。

## Drawable实现细节

由于屏幕视口随时变化，故每次绘制前都会进行屏幕矩形检查防止画面压缩变形（screenViewpMatirx），与活动块Drawable的Transform类的矩阵进行乘算（transMatrix），

具体顶点着色器实现：

gl\_Position = screenViewpMatirx \* transMatrix \* vec4(position, 0, 1);

其中screenViewpMatirx计算方式：

[a, 0, 0, 0,

0, b, 0, 0,

0, 0, -2, 0,

-1, -1, -1, 1]

a = 2 / <width>

b = 2/ <height>

transMatrix计算方式：

参考Simple and Fast Multimedia Library的Transform计算方式。

需要注意每次进行DrawElements操作前都要确保GL的Viewport正确设置。

## 线程模型参考

事件线程（主线程）

脚本执行线程

图像渲染线程

辅助线程（音频服务，并行计算等）

## 核心架构设计

计划将内核代码分为以下部分：

1. 事件线程处理
2. 脚本渲染线程
3. 图像渲染与OGL线程
4. 音频及其他并行计算线程

## 周边设施

UI及Widget渲染创建 -> SDL\_Window SDL\_Surface 软渲染软件内置UI

音频解码及视频多媒体解码 -> 计算解码部分

字体管理器，根据特定平台特化字体处理流程