# 修改配置参数

## 需求：

1、想把安卓机的可视化距离r.ViewDistanceScale，从0.5改为0.65

## 执行：

1. 修改BaseDeviceProfiles.ini 内安卓的grade01~03的r.ViewDistanceScale到0.65；
2. 拉分支修改后，提交推送后并MR

## 小结：

1. BaseDeviceProfiles.ini，存一些根据机型档次、画质等级而修改的控制台变量 CVar  
   BaseFastPictureSwitch.ini，存一些启动之后就不会修改的 CVar  
   BaseFastPictureMatch.ini，机型匹配的规则
2. 安卓低端机：Android\_Grade01-03

iOS低端机：iOS\_Smooth

3、git的使用，参考：

<https://backlog.com/git-tutorial/cn/>

以及goodnotes

# 三个优化问题

## 需求：

1. 尝试优化以下结构：

static FUObjectAnnotationDense<FLinkerIndexPair,false> LinkerAnnotation;  
static FUObjectAnnotationDense<FArchetypeInfo, true> ArchetypeAnnotation;   
mutable TMap<FName, UFunction\*> SuperFuncMap;

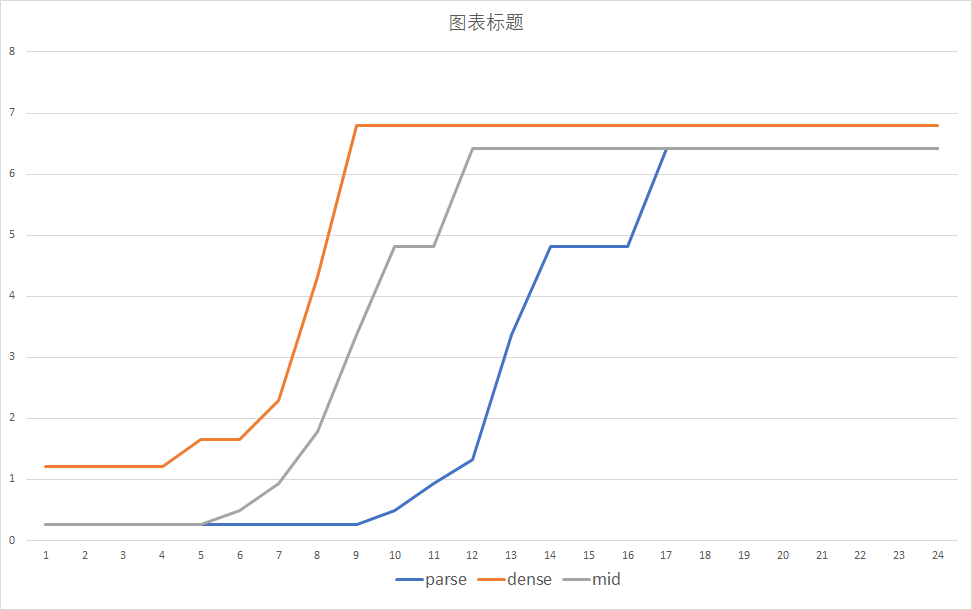
## 执行：

1. 细读代码，尝试理解其运行原理
2. 使用LLM进行内存跟踪，参考：

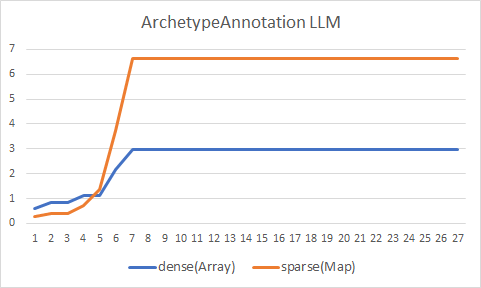
<https://docs.unrealengine.com/zh-CN/Programming/Development/Tools/LowLevelMemoryTracker/index.html>

<https://wangjie.rocks/2019/08/13/ue4-llm/>

3、其中第一个可将本身使用TArray改为TMap（内存轻微下降），内部已经写好了，修改一处即可：



第二个和第三个已经用的是TMap，无优化空间：



# 屏蔽背景模糊问题

## 需求：

1. 屏蔽非HDR的登录界面等地方弹出窗口时，背景的模糊效果

## 执行：

1. 运行游戏，到背景模糊处，用UE4自带工具，定位到背景模糊的UI（Widget）蓝图，找到这个蓝图里面所用的组件
2. 定位组件名，到底层CPP里面去修改
3. 发现这个组件底层CPP已经写了屏蔽开关（且已有控制台变量），所以任务变成了配置参数即可
4. 在BaseDeviceProfile.ini 里面将所有非HDR添加参数Slate.AllowBackgroundBlurWidgets =0，所有HDR添加参数Slate.AllowBackgroundBlurWidgets=1

## 小结：

1. 用UE4定位插件的功能：Window-Develop Tools-Widget Reflector 进行Pick
2. 用控制台变量来传给CPP的功能，并配置参数

# 修改配置参数

## 需求：

1、将r.Mobile.AllowSoftwareOcclusion 这个配置等于2 的全都改成1

## 执行：

1、拉分支，在BaseDeviceProfile.ini 里面将所有r.Mobile.AllowSoftwareOcclusion=2的全部改为1，并提交推送MR

## 小结：

# 冒烟日志LOG ERROR

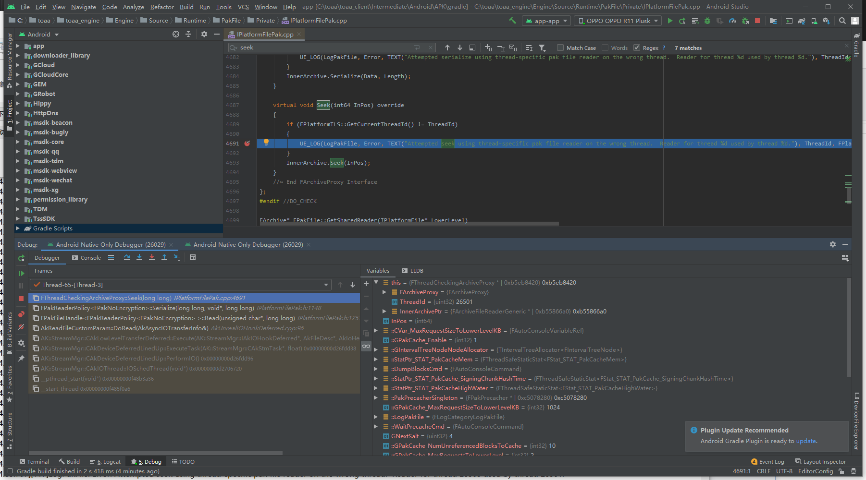
## 需求：

1、

|  |
| --- |
| LogPakFile: Error: Attempted seek using thread-specific pak file reader on the wrong thread.  Reader for thread 17211 used by thread 17209. |
| LogPakFile: Error: Attempted serialize using thread-specific pak file reader on the wrong thread.  Reader for thread 17211 used by thread 17209. |

## 执行：

1. 用Android Studio进行断点，发现这个Log出现在：



可能原因：音频文件与其他文件在同一个包里，在音频读取之前，这个包就已经被打开了，由于读取音频是在另外一个线程，就会导致多个线程读取同一文件的错误

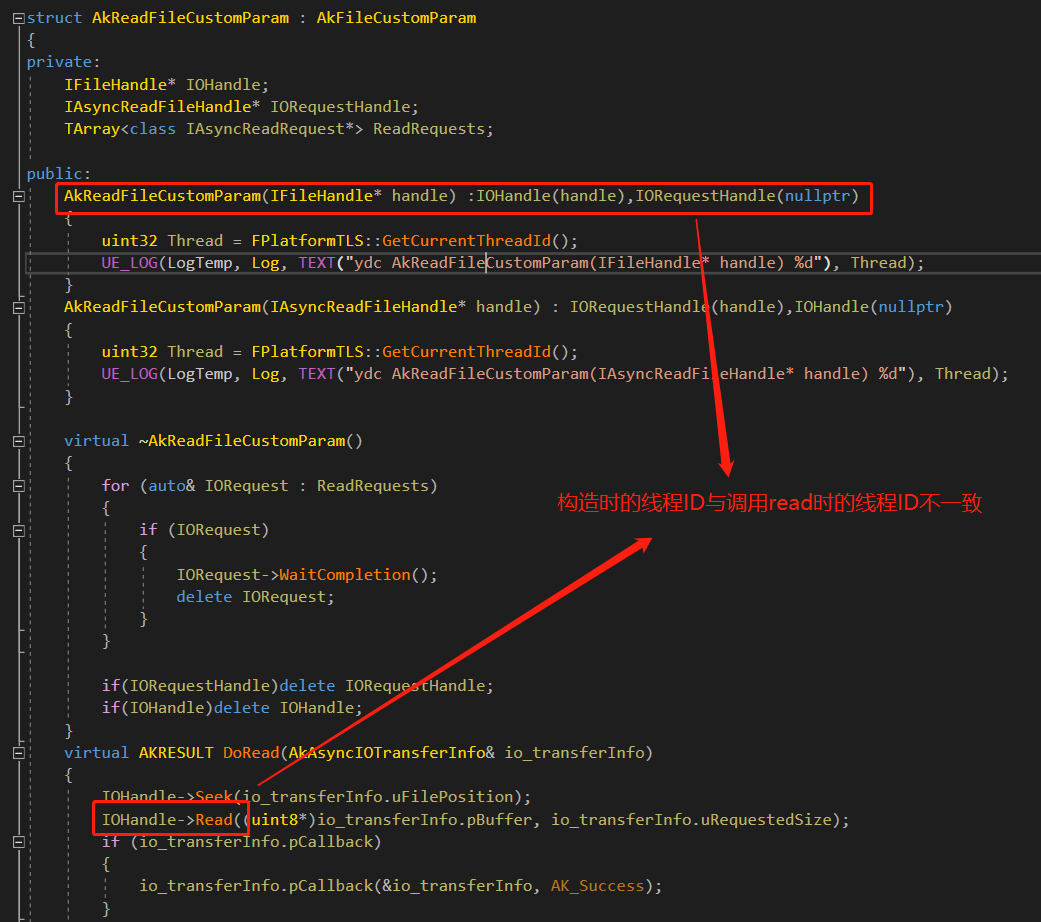
修改：将音频文件分割出来打一个包

反馈：仍然有类似问题

1. 可能原因：尽管分割出一个包，但是这个包仍然在读取音频之前就打开了

查询：确认以上

确认结果：发现在，构造和调用时线程不一致，但是构造和调用发生在系统函数里面，改不了



修改：在read时加锁，并在open 的时候将线程保存到一个全局变量，然后在check的时候根据当前线程是否在音频线程确定是否check

结果：加锁后加载时间增加了5s，复测几次后发现时间无明显变化，在oppo r11上面进入大世界均为87s

## 小结：

1. 安卓环境配置和安卓打包、adb的使用以及断点调试，参考：

<http://tapd.oa.com/Xgame2/markdown_wikis/#1020378222009323755>

及Goodnotes

2、ue4加锁的操作，参见另外一个文档：ue4加锁的操作.docx

# 冒烟日志LOG ERROR

## 需求：

1、

|  |
| --- |
| LogStreaming: Error: Found 0 dependent packages... |
| LogStreaming: Error: This will hitch streaming because it ends up searching the disk instead of finding the file in the pak file. |

## 执行：

1. 安卓断点调试，初步发现：

可能原因：在ImageDownloader里面，会按照从内存到硬盘再到服务器的顺序下载图片，如果第一次没从内存下载，则会从硬盘或者服务器下载，并把文件添加一个句柄到Map里面，标记已经在内存里了，但是在切换场景的时候，清理了内存却没有清理这个Map，导致再次下载这张图片的时候，就会首先进入内存里面去查找，但是查找失败，就会报这个Log

修改：让切换场景的时候，把Map给清理掉，考虑在MapTransferMgr:ClearPreviousMap里面调用一个CPP的方法，CPP方法写在UImageDownloader里面

1. 修改：在UImageDownLoader里写了一个成员函数ClearHashPathMap来清理

HashPathMap，在MapTransferMgr:ClearPreviousMap来调用这个函数

验证：上述修改的确会消掉那个Log error

## 小结：

1. 如何在lua里面调用cpp函数：

在.h 文件里面声明函数为static，并在结尾ADD\_STATIC\_FUNCTION( XXX )；

在.cpp文件里面定义此函数

在lua里面用 类名.函数名 的形式调用

# BUG单 设置帧率不生效的问题

## 需求：

1、在IOS机器上面，设置帧率不生效

## 执行：

1、尝试在安卓机测试，没有复现

2、没有IOS机器，暂停开发

3、在ios上面用命令行参数设置发现未生效，转单给知晓

小结：

1、t.maxFPS 控制最大fps

2、r.vsync 开启关闭垂直同步，rhi.syncinterval 控制垂直同步间隔

3、 垂直同步的概念：为了避免屏幕刷新率低于游戏的帧率而出现的撕裂现象，所以等待屏幕逐行显示完毕后，再画下一帧

# BUG单 设置阴影不生效的问题

## 需求：

1. 在设置关闭阴影之后，传送场景，阴影会出现，IOS、安卓、PC上面均会复现

## 执行：

1. 搜索关键词SetShadow、Shadow等定位到相关位置
2. 初步假设：没有存储阴影参数，或者切换场景时冲掉了这个参数

尝试：在MapTransferMgr的EnterMap函数里面添加了“获取shadow，设置shadow”的语句，发现在编辑器里面，设置阴影恢复了正常

3、进一步的，发现是UI系统组的逻辑问题，转给littlema

# HUAWEI P20 pro默认配置查询

## 需求：

1. 测试那边报了一个问题：HUAWEI P20 pro 默认应该是 High（其实应该是Balance），但实际上是 HighHDR。  
   需要你这边帮忙查一下  
   https://docs.qq.com/sheet/DWlJJckZVcWlISVNS?tab=BB08J2  
   这个是我们根据公司的 TDR v.21 标准，整理出来的 GPU 对应档次文档

## 执行：

1. 根据TDR v.21标准表格，将BaseFastPictureSwitch.ini里面所有机型核查修改了一次。

# 不发布Xhook代码

## 需求：

1. 不让xhook代码出现在shipping包里面

## 执行：

1. 在所有xhook相关代码开头加上宏 **!UE\_BUILD\_SHIPPING** 进行限制
2. 打包后，解析.so文件，看是否有xh或xhook相关关键字，事实证明有效

## 小结：

1、.so文件的解释：<https://www.jianshu.com/p/ef462437b999>

2、.so文件的解析：linux系统下可直接用nm\objectdump\readelf\ldd等命令（可见<https://stackoverflow.com/questions/34732/how-do-i-list-the-symbols-in-a-so-file>）而在windows系统下，则需要先下载cygwin，添加binutils包，再使用这些命令