Logger使用说明

1. 在目录build.gradle下

implementation project(":Logger\_harmony")

1. 在项目的MyApplication->onInitialize方法中添加如下代码，Log级别必须是debug以上、

Logger.*addLogAdapter*(new HarmonyOsLogAdapter() {  
 @Override  
 public boolean isLoggable(int priority, @org.jetbrains.annotations.Nullable String tag) {  
 return BuildConfig.*DEBUG*;  
 }  
});

1. 基本用法

Logger.*i*("需要打印的日志信息");

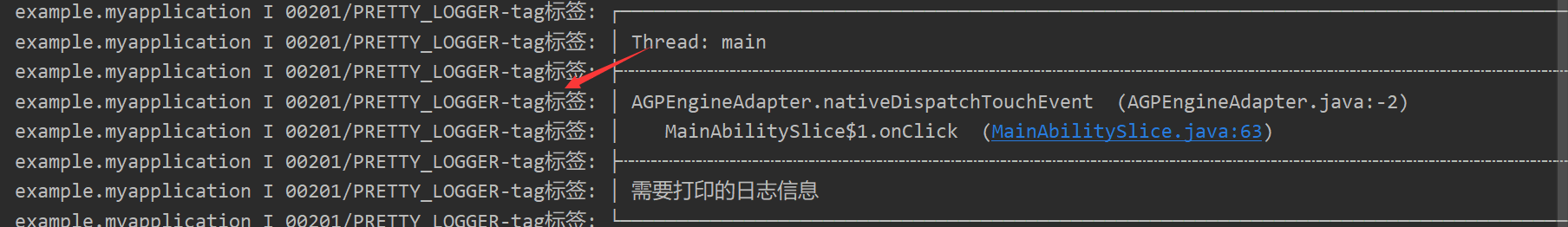
1. 输出结果,没有tag标签



1. tag标签用法

Logger.*t*("tag标签").i("需要打印的日志信息");

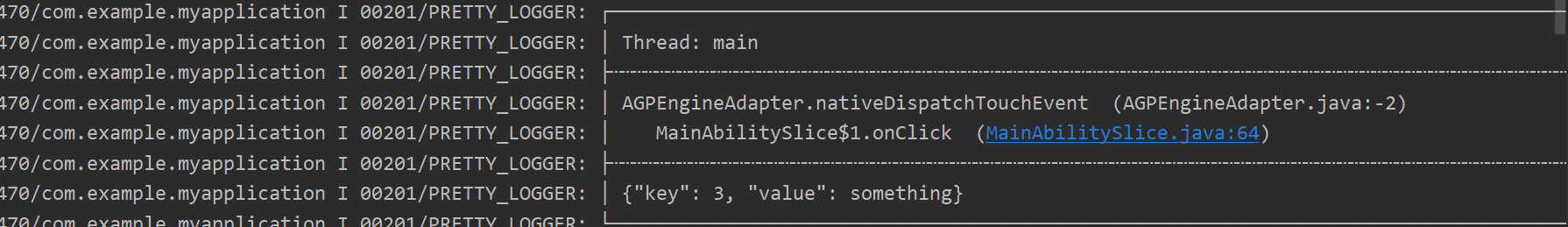
1. 输出结果



1. Json用法

Logger.*json*("{\"key\": 3, \"value\": something}");

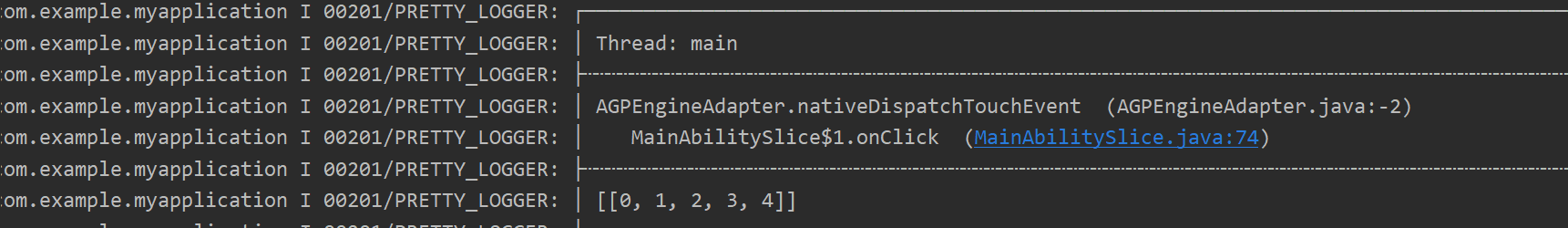
1. 输出结果



1. List用法

List<Integer> list = new ArrayList<>();  
for (int i = 0; i < 5; i++) {  
 list.add(i);  
}  
Logger.*i*(Arrays.*asList*(list));

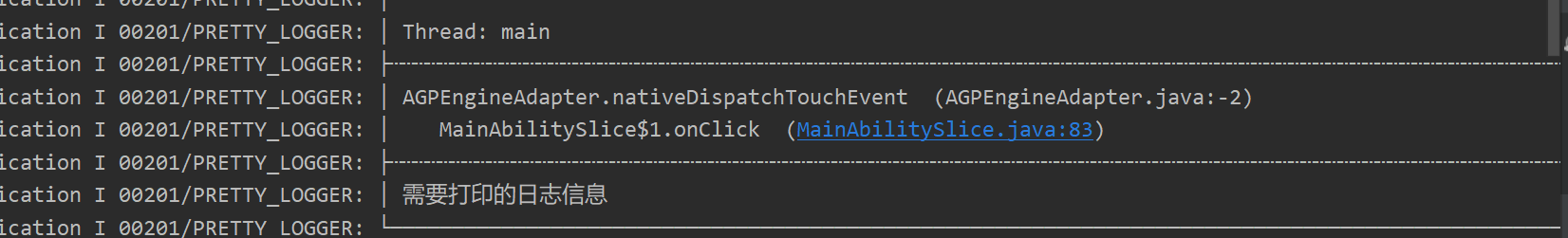
1. 输出结果



1. 进阶用法,可以打印出事件分发方法.

FormatStrategy formatStrategy = PrettyFormatStrategy.*newBuilder*()  
 .showThreadInfo(false) // (Optional) Whether to show thread info or not. Default true  
 .methodCount(0) // (Optional) How many method line to show. Default 2  
 .methodOffset(7) // (Optional) Hides internal method calls up to offset. Default 5  
 .logStrategy(new LogcatLogStrategy()) // (Optional) Changes the log strategy to print out. Default LogCat  
 .tag("My custom tag") // (Optional) Global tag for every log. Default PRETTY\_LOGGER  
 .build();  
  
Logger.*addLogAdapter*(new AndroidLogAdapter(formatStrategy));  
Logger.*i*("需要打印的日志信息");

1. 输出结果



1. 日志保存到分布式目录中,调用方法

Logger.*addLogAdapter*(new DiskLogAdapter(context));

1. 分布式目录创建方式.

File distDir = context.getDistributedDir();  
String filePath = distDir + File.*separator* + "logger.csv";

1. 目前日志之支持4种级别打印,等待harmonyOs Api后续更新

*INFO ,WARN,ERROR,ASSERT*