饿了么

饿了么物流技术交流群

扫描二维码加入>>>

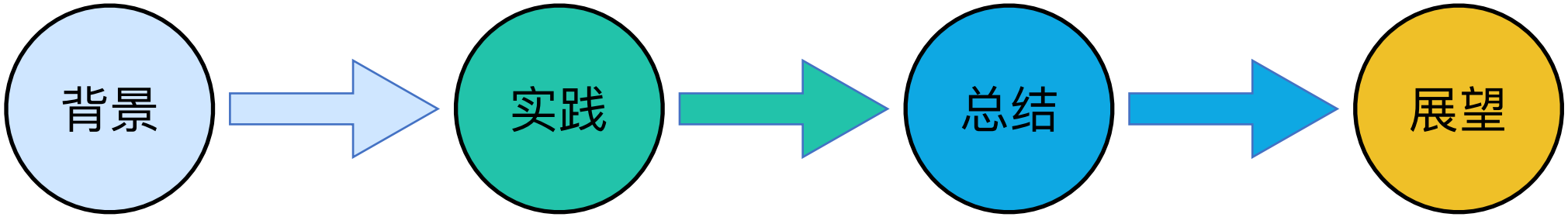
蜂鸟团队移动端异常监控体系建设

主讲人：潘万坤

个人简介

潘万坤

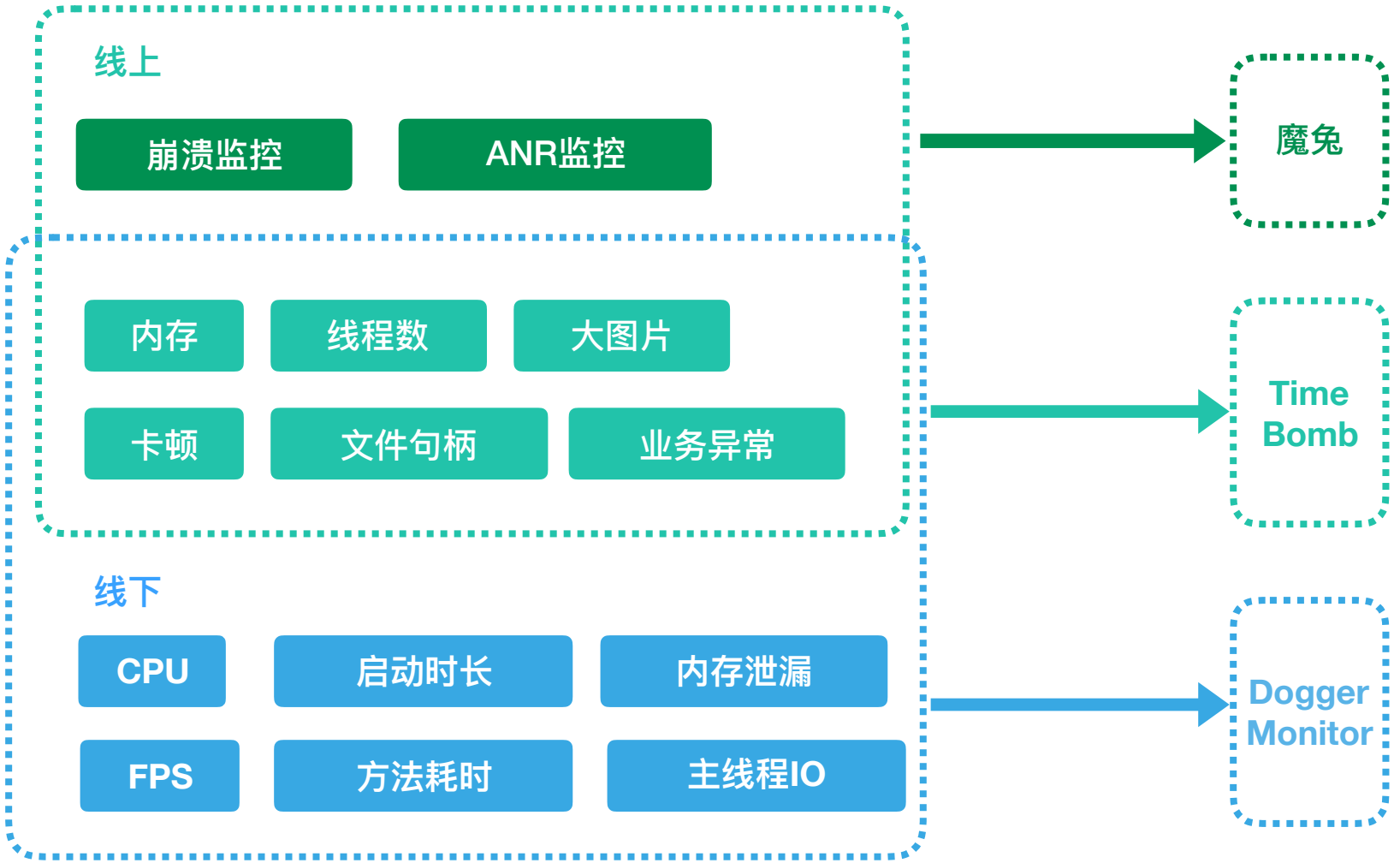
2013年毕业于武汉大学，之后在格瓦拉、饿了么等公司从事安卓开发工作。目前是饿了么物流移动团队安卓负责人，主要关注移动端代码架构、性能优化等领域，负责保障App在线上的高稳定运行。


背景
实践
总结
展望

背景

01	崩溃、ANR等异常情况太单一
02	线上环境复杂，复现难度高
03	提高发版质量的要求

实践


线上
崩溃监控
ANR监控
内存
线程数
大图片
卡顿
文件句柄
业务异常
线下
CPU
启动时长
内存泄漏
FPS
方法耗时
主线程IO
魔兔
Time Bomb
Dogger Monitor

实践 – TimeBomb (自定义异常监控系统)

实践 – TimeBomb (自定义异常监控系统)

含义： 一段时间内连续出现N次的异常

示例： 登录异常

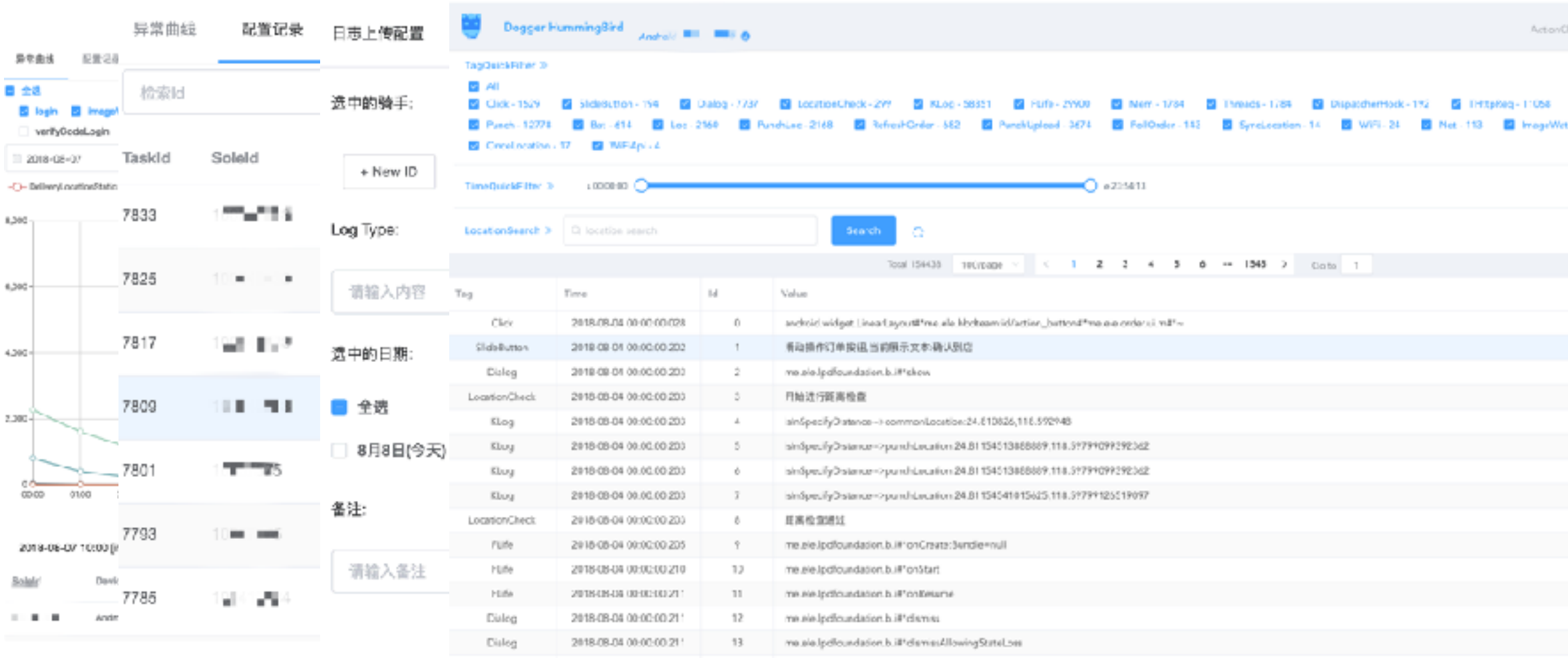
送单异常

定位异常

大图片申请

卡顿

实践 – TimeBomb


异常曲线
配置记录
日志上传配置
检索Id
TaskId
SoleId
+ New ID
7833
7825
7817
7809
7801
7793
7785
选中的骑手:
Log Type:
请输入内容
选中的日期:
全选
8月8日(今天)
备注:
请输入备注
DaggerHummingBird
Search
Tag
Time
Id
Value

实践 – Dogger日志系统

Life	Click	Http	Custom
生命周期	点击	网络请求	自定义

实践 – Dogger日志系统

特点

01	mmap写文件，高效
02	远端配置，及时上传
03	压缩比高，省流量
04	敏感信息加密，安全
05	可生成自动化脚本，功能强大

实践 – Dogger日志系统

开源！！！

地址：https://github.com/**eleme/Trojan**

实践 – DoggerService

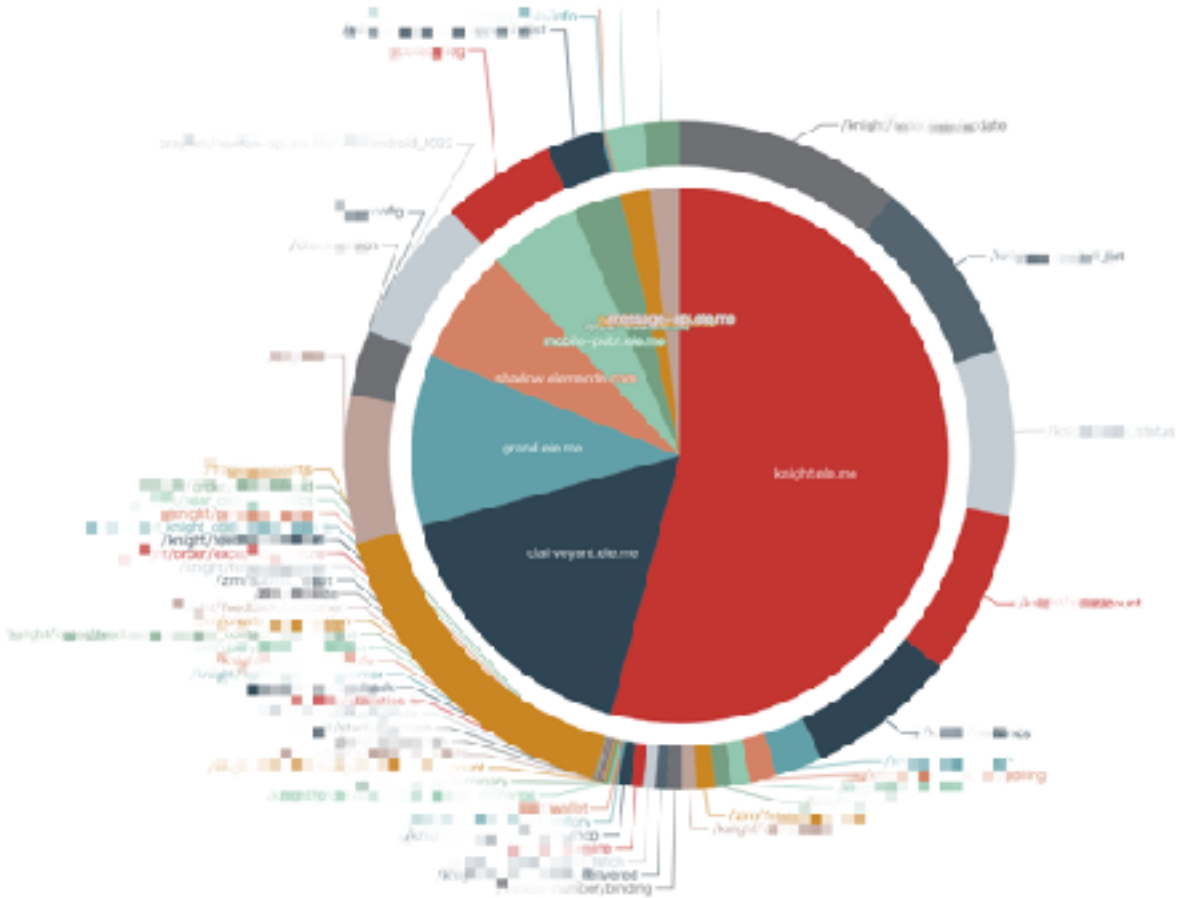
小目标

01 优雅地展示日志内容

02 对埋点的事件频率做直观的展示

03 对特定的业务模块有一定的数据分析能力

实践 – DoggerService



实践 – DoggerMonitor

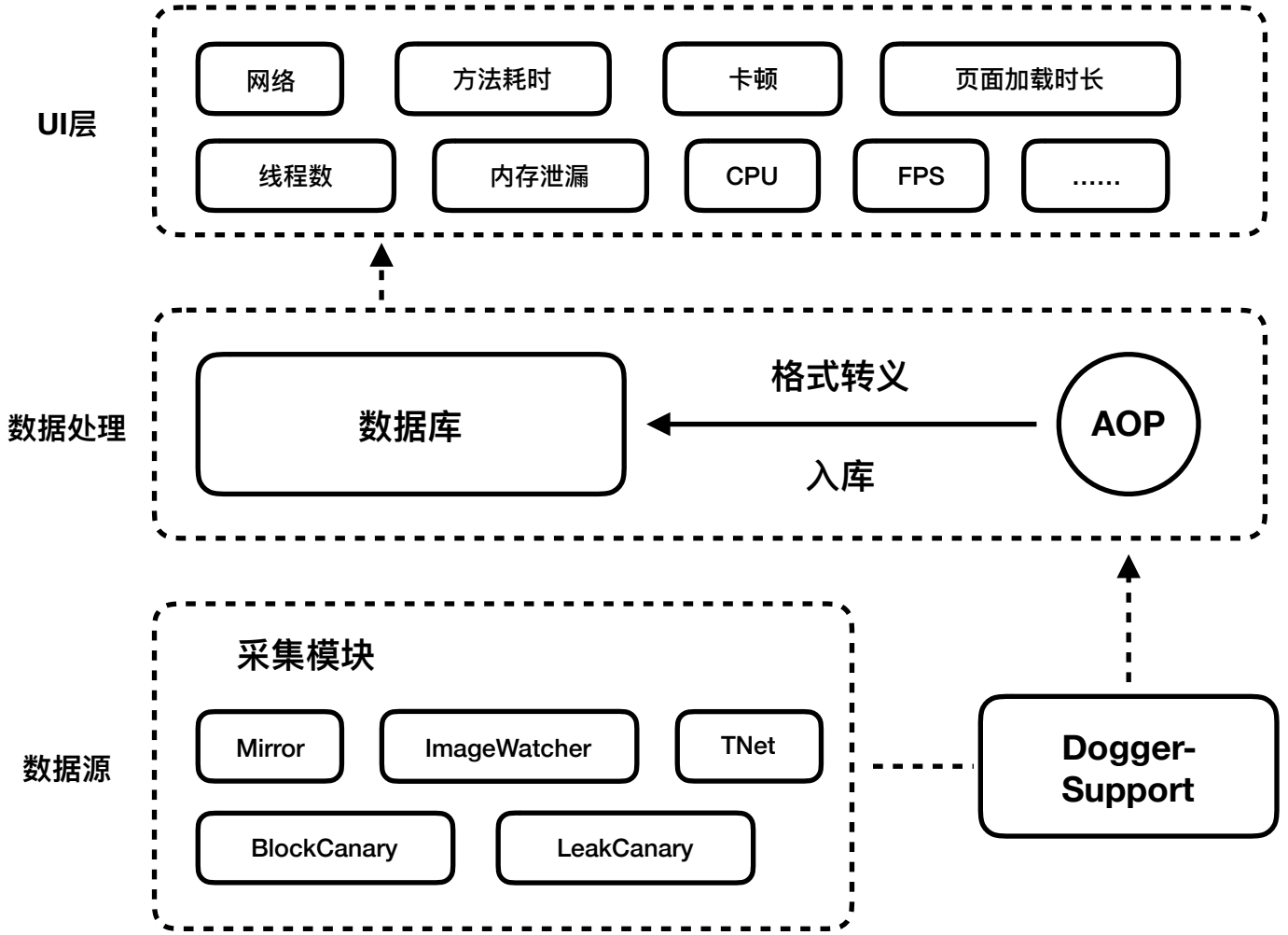
背景

01	线上性能优化成本高
02	线下监控可以牺牲部分性能
03	谁引入谁治理的方式保证质量

实践 – DoggerMonitor开发体验

实践 – DoggerMonitor

架构图



实践 – DoggerMonitor实现细节

方法耗时 - Mirror

```
void method(){
    long startTime = System.currentTimeMillis();
    ...
    ...
    long endTime = System.currentTimeMillis();
    MirrorUtils.mirror(endTime - startTime, className + methodName + params);
}
```

实践 – DoggerMonitor实现细节

卡顿 - BlockCanary

```
Choreographer.getInstance()
    .postFrameCallback(new Choreographer.FrameCallback() {
        @RequiresApi(api = Build.VERSION_CODES.JELLY_BEAN)
        @Override
        public void doFrame(long l) {
            BlockCanaryManager.getInstance().stop(runnable);
            BlockCanaryManager.getInstance().start(runnable , TIME_BLOCK);
        }
    });
```

实践 – DoggerMonitor实现细节

流量统计 - TNet

```
public class TNetInterceptor implements Interceptor {
    @Override
    public Response intercept(Chain chain) throws IOException {
        Request request = chain.request();
        Response response = chain.proceed(request);

        recordHttp(request);
        recordHttp(response);

        return response;
}
```

实践 – DoggerMonitor实现细节

流量统计 - TNet

```
@TargetClass("okhttp3.OkHttpClient")
@NameRegex("okhttp3/RealCall")
@Proxy("networkInterceptors")
public List<Interceptor> networkInterceptors() {
    List<Interceptor> interceptors = (List<Interceptor>) Origin.call();
    List<Interceptor> newList = new ArrayList<>(interceptors.size() + 1);
    newList.addAll(interceptors);

    newList.add(new TNetInterceptor());

    return newList;
}
```

实践 – DoggerMonitor实现细节

流量统计 - TNet

BUILD_VERSION >= 23

```
NetworkStatsManager.querySummary(ConnectivityManager.TYPE_MOBILE,
        uid,
        startTime,
        endTime);
```

BUILD_VERSION < 23

```
TrafficStats.getUidRxBytes(uid) + TrafficStats.getUidTxBytes(uid);
```

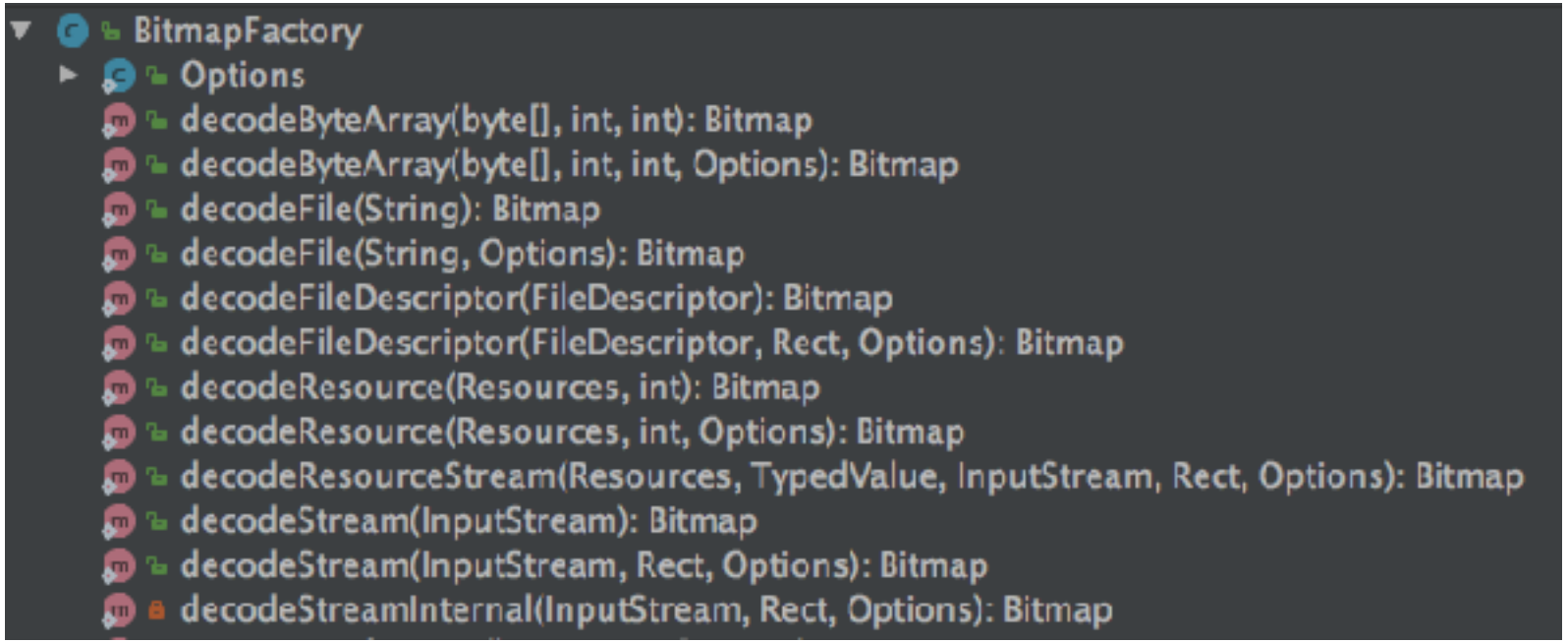
实践 – DoggerMonitor实现细节

大图片检测 - ImageWatcher

```
public int computeSize(int targetDensity) {
    if (density == Image.DEFAULT) {
        return width * height * 4;
    } else {
        return (int) ((width * targetDensity / density + 0.5f)
                * (height * targetDensity / density + 0.5f)
                * 4);
    }
}
```

实践 – DoggerMonitor实现细节

大图片检测 - ImageWatcher



总结

01	线上异常的及时上报
02	提高问题追查的效率
03	保障开发版本的质量

展望

01 DoggerMonitor自动化测试流程化

02 DoggerMonitor后台管理系统

Q&A

饿了么

饿了么
物流
技术交流群

扫描二维码加入>>>

Thanks !