APMCon

手机卫士性能优化方案

2017年8月

内容列表

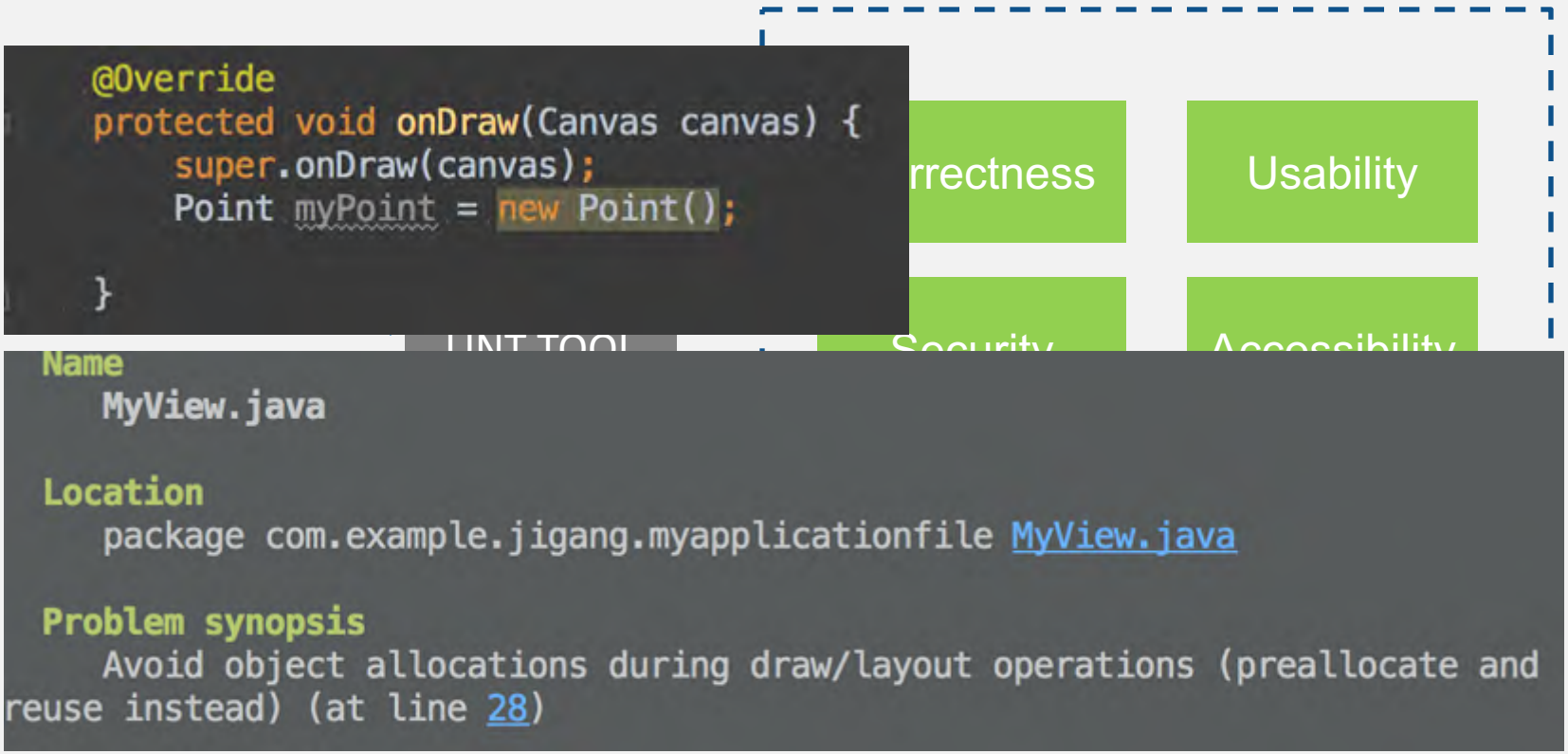
APMCon

- 预防性能问题
  - 代码检测工具
  - 自动化分析
- 性能监控
- 问题诊断
  - 资源优化工具
  - APK构建检测
- 修复技术

代码检测工具

APMCon

LINT扫描神器



代码检测工具

APMCon

FINDBUGS——深度定制

警告类型
NP
OS
LSYC
AUC
SPP
FCBL
QIHOO_RL
RCN

```
Cursor cursor = ContextHelper.getMainContext().getContentResolver().query(
try {
    if (cursor != null && cursor.moveToFirst()) {
        do {

        } while (cursor.moveToNext());
    }
} catch (Exception e) {
    LogUtils.logE(e);
} finally {
    IoStreamUtils.closeQuietly(cursor);
}
```

原生findbugs认为有问题

```
if (cursor != null) {
    try {
        cursor.close();
    } catch (Throwable e) {
        if (DEBUG) {
            Log.e(TAG, e.getMessage(), e);
        }
    }
}
```

代码检测工具

APMCon

代码火线—不可穿越的火线

UI

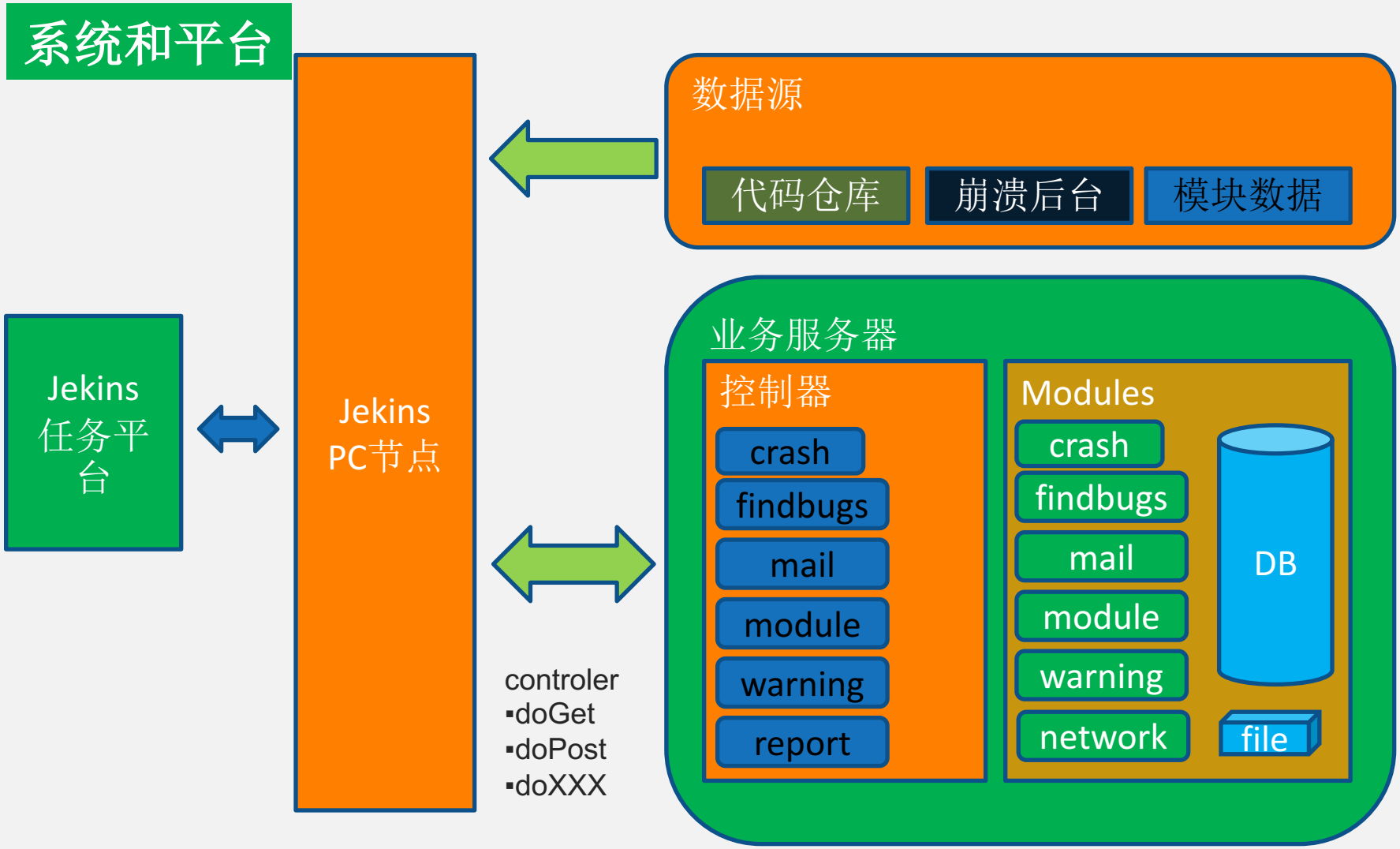
- UI线程的耗时操作
- 频繁查找资源
- 频繁创建对象
- 过度刷新
- Adapter复用
- 布局扁平
- 直接使用大图片

Perf

- Dex加载
- 解压
- 数据延迟加载
- 后台任务调度
- Webview

自动化分析

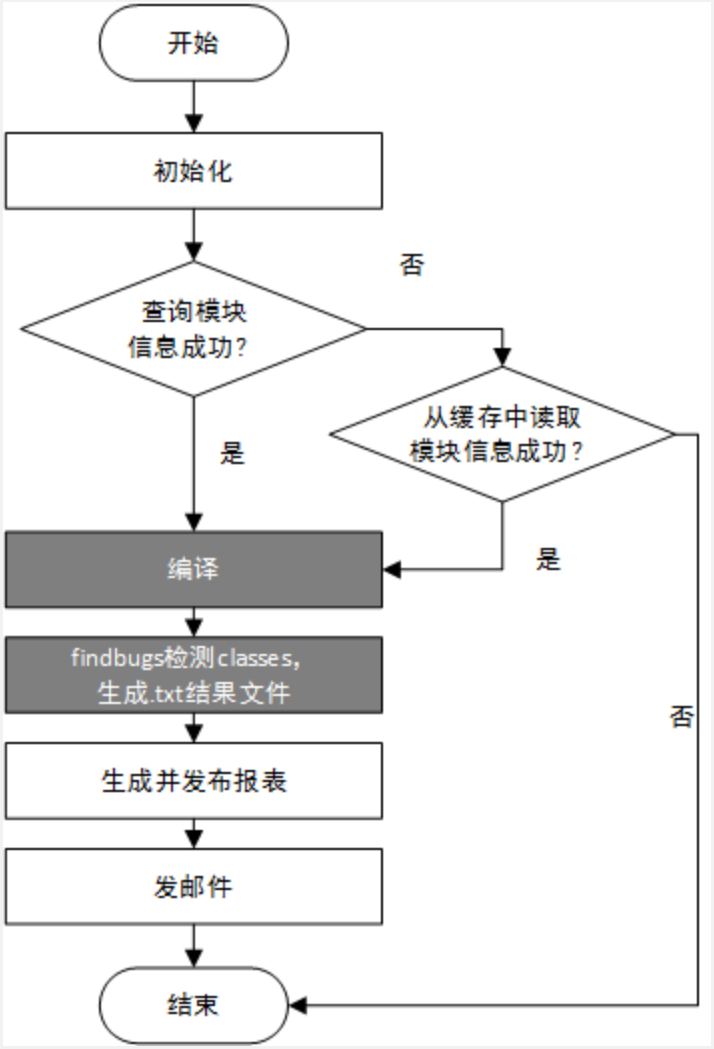
APMCon



自动化分析

APMCon

举例



输出

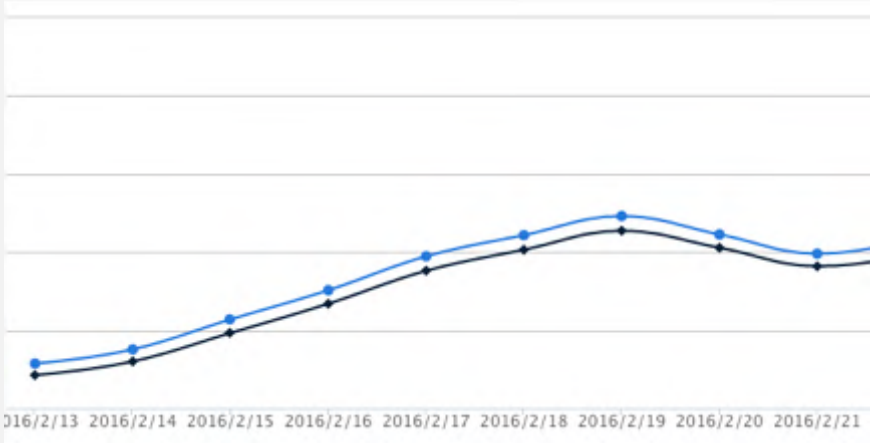
Hi,

请检查下面的 findBugs 问题:

Time\Type	NP	OS	
2016/03/14	0	1	

DETAIL: http://xxx.xxx.xxx/view/CodeStaticCheck/job/AutoFindbugs/205/label=
PS：没有账号的可以通过公共账号登录：用户名 xxx，密码 xxx

警告类型	类型说明
NP	可能存在空指针
OS	IO流可能未能正常关闭
LSYC	可能在局部变量的中，不必要的使用线程安全类，如StringBuffer等
AUC	CURSOR 可能没有正常关闭
SPP	在使用"instanceof"之前使用"==null"判空，没必要，instanceof本身就会判空
FCBL	定义了域，实际只在一个方法里面使用，请将该域对应的变量替换成方法里面的局部变量
QIHOO_RL	文件流或者指针关闭不正确
RCN	1. Nullcheck of… 表示在判断变量为空前，使用了该变量
	2. Redundant nullcheck of… 表示判空是多余的



APK构建

APMCon

重复资源　图片压缩　资源混淆　核心法宝：制度

build	build	build	build	build	包大小 方法数	包大小查看 (debug)	包大小查看 (release)	查看 FindBugs
debug	findbugs	strict debug	release	status	detail	detail	detail	detail

版本,编译号	日期,时间	修订	包大小
6.5.1.1032	2016-03-14 20:02:46	1558442	10154328 -779 ↓
6.5.1.1031	2016-03-14 19:28:54	1558256	10155107 -4835 ↓
6.5.1.1031	2016-03-14 15:49:33	1557914	10159942 +60267 ↑

字符串数	类型数	原型数	字段数
18302	4089	5231	10019
18302 -42 ↓	4089	5231	10019 -44 ↓
18344 -1049 ↓	4089 -265 ↓	5231 -296 ↓	10063 -1567 ↓

方法数	类数	字节码大小
25163	3275	2631788
25163 +6 ↑	3275	2631788 -1020 ↓
25157 -1382 ↓	3275 -235 ↓	2632808 -141504 ↓

下一个

- 预防性能问题
- 性能监控
  - 埋点收集信息
  - 常规性能监控
  - 观察者工具
  - 性能分析工具
- 问题诊断
- 修复技术

Aspectj重写器

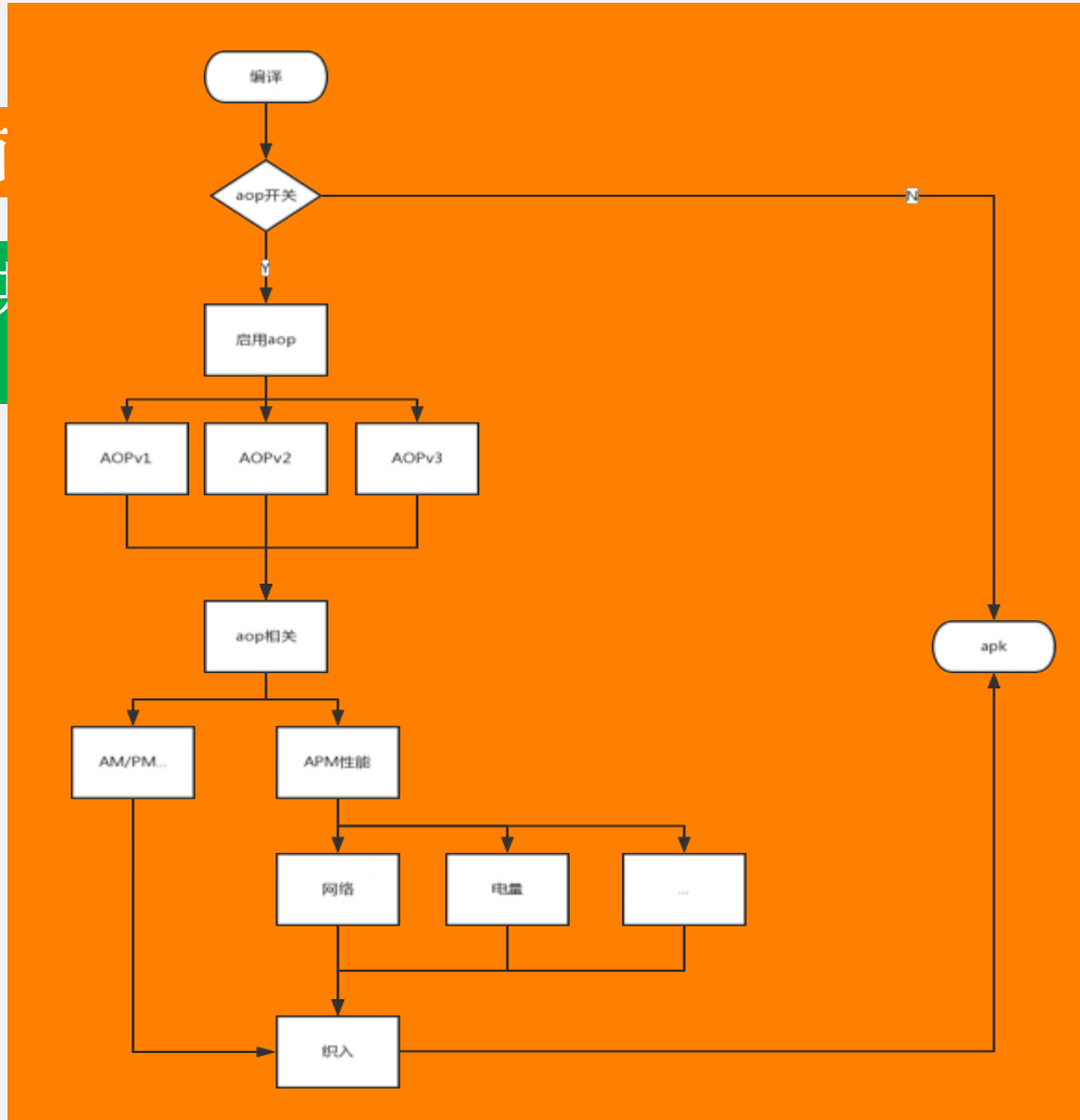
介绍及应用

基于java的面

静态植入：编译时
文件中完成织入

电量

网络交互



常规性能监控

APMCon

- 预防性能问题
- 性能监控
- 问题诊断
- 修复技术

性能监控：
- 埋点收集信息
- 常规性能监控
  - 进程冷热启动
  - 四大组件启动速度
  - 各个线程卡慢检测
  - 异步处理性能检测
- 观察者工具
- 性能分析工具

进程的冷热启动

APMCon

进程冷热启动

Context::startXXXX
Context::bindXXXX
Application::attach
Application::onCreate
Activity:onCreate

CustomInstrumentation::callApplicationOnCreate

service: exported?
activity: exported? UI?
Provider: exported?
Receiver: exported?

四大组件启动

APMCon

举例—Activity的加载时间

关键字：
ActivityThread
Instrumentation

CustomInstrumentation

```
public void callActivityOnCreate(Activity activity, Bundle icicle) {
    //...
    //收集我们需要的activity信息。
  long bTim = System.currentTimeMillis();
   rTRunnable.setBeginTime(bTim , activity.getComponentName().getClassName());
    super.callActivityOnCreate(activity, icicle);
    //activity绘制完成后，执行回调函数，用来记录时间
    activity.getWindow().getDecorView().post(rTRunnable);
    //...
}
```

className	timeType	time	day	flag
com.qihoo360.mobilesafe.ui.index.AppEnterActivity	1	375	16794	0
com.qihoo360.mobilesafe.paysafe.nicaifu.NicaifuMain	1	373	16794	0
com.qihoo360.mobilesafe.block.ui.BlockRecordScreen	1	122	16794	0
com.qihoo360.mobilesafe.block.ui.BlockRecordScreen	1	252	16794	0
com.qihoo360.mobilesafe.main.ui.AppManagerActivity	1	150	16794	0

线程卡慢监控

APMCon

采样时间发送检测信息

Handle::post
Handle::dispatchMessage

CustomInstrumentation::waitForIdle
CustomInstrumentation::waitForIdleSync

```
Public void CustomInstrumentaion::waitForIdle(){
//…
mMessageQueue.addIdleHandler(new Idler(recipient));
mThread.getHandler().post(new EmptyRunnable(t));
//…
}
```

异步处理的性能监控

APMCon

异步处理静态埋点

拦截对象：

Handle::post(Runnable r)
Handle::postAtTime(Runnable r, long uptimeMillis)
Handle::postAtTime(Runnable r, Object token, long uptimeMillis);
Handle::postDelayed(Runnable r, long delayMillis)
Handle::postAtFrontOfQueue(Runnable r)

常规性能监控

APMCon

- 预防性能问题
- 性能监控
  - 埋点收集信息
  - 常规性能监控
    - 进程冷热启动
    - 四大组件启动速度
    - 各个线程卡慢检测
    - 异步处理性能检测
  - 观察者工具
  - 性能分析工具
- 问题诊断
- 修复技术

卡慢

- Java 反射
- 四大组件的onCreate执行时间
- ClassLoader.getResourceAsStream
- 第三方SDK

工具

APMCon

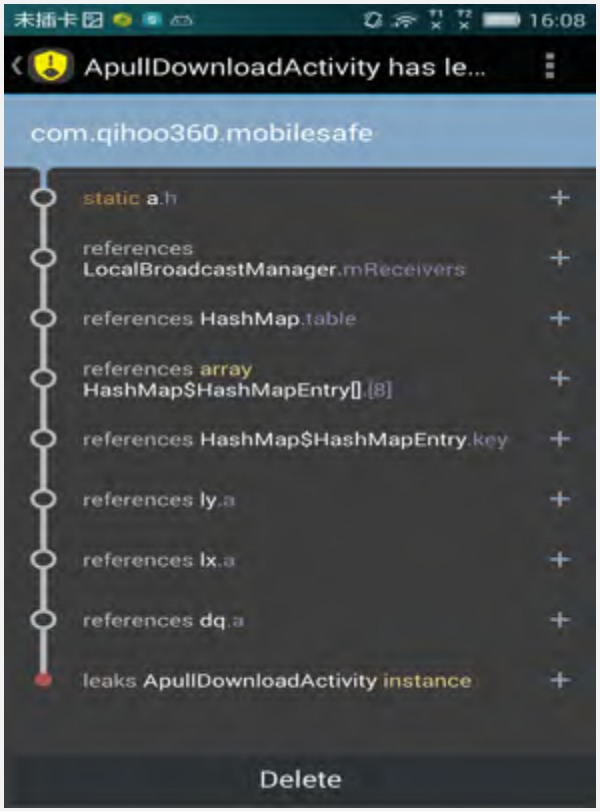
- 预防性能问题
- 性能监控
  - 埋点收集信息
  - 常规性能监控
  - 观察者工具
  - 性能分析工具
    - LeakCanary
    - 分析工具（例：内存）
    - APM集成工具——Argus APM
- 问题诊断
- 修复技术

观察者App

APMCon

LeakCanary

检查Activity是否泄漏的问题，通过onDestroy方法检查，存在泄漏，则会dump内存镜像，获取栈信息



内存分析

APM

分析方法

- Dalvik H
- adb shell kil
- Debug.dum
- DDMS dum
- hprof-conv

基本信息

HPROF文件	对象数量	GC Roots数量	Class数量	ClassLoader数量	已用Dalvik堆内存 (byte)	卫士对象数量	卫士对象内存 (byte)	可回收对象数量	可回收内存 (byte)
init.hprof	70608	14623	4941	10	18039568	861	1253680	5676	208424
jiaozhun.hprof	74902(+4294)	15814(+1191)	5243(+302)	11(+1)	18286448(+246880)	988(+127)	1338840(+85160)	887(-4789)	49040(-159384)

* 对象数量是指在该Hprof文件中产生的对象总数，代表了在dump hprof文件时该进程中的java层对象总数，同理GC Roots数量，Class 数量和ClassLoader数量
* 已用Dalvik堆内存代表该进程在dump hprof时占用的Dalvik 堆内存大小，已用Dalvik堆内存 = 卫士程序占用内存 + Android系统对象占用内存 + 可回收内存
* 卫士对象数量和卫士对象内存代表由卫士程序定义的类而产生的对象数量和这些对象占用的Dalvik堆内存大小
* 可回收对象数量和可回收对象内存代表在dump hprof时，该进程Dalvik堆内存中可以被回收的对象总数和内存大小

* 下表展示了Hprof文件中卫士程序自定义的类及其对象的内存占用情况

详细信息

名称	init.hprof			jiaozhun.hprof		
	对象数量	自占内存 (byte)	持有内存 (byte)	对象数量	自占内存 (byte)	持有内存 (byte)
com.qihoo360.mobilesafe.notification.NotificationService$Inner	1	48	696720	1	48	696720
com.qihoo360.mobilesafe.paysafe.modules.AppVerify	1	24	70608	1	24	70808(+200)
com.qihoo360.mobilesafe.paysafe.db.AppVerifyDatabase	1	16	70336	1	16	70336
com.qihoo360.mobilesafe.paysafe.db.AppVerifyDatabase$PayRule	250	8000	64192	250	8000	64192
com.qihoo360.mobilesafe.block.data.YellowNumberManager	0	0	59552	0	0	59552
com.qihoo360.mobilesafe.share.NumberManager	0	0	42704	0	0	42856(+152)
com.qihoo.antivirus.shield.domain.ShieldDisguiseLocationInfo	20	1120	37136	20	1120	37136
com.qihoo360.common.utils.IniProperties	1	16	8448	2(+1)	32(+16)	27880(+19432)
com.qihoo360.loader2.Plugin	14	560	25960	14	560	27656(+1696)
com.qihoo.antivirus.shield.domain.PackageItem	79	9480	12040	79	9480	12336(+296)
com.qihoo360.mobilesafe.block.provider.SafeGuardProvider	1	48	10328	1	48	10328
com.qihoo360.mobilesafe.paysafe.payrecord.TriggerFactory	1	24	9048	1	24	9048
com.qihoo.antivirus.autostart.AutoStartedPackageItem	63	3024	8600	63	3024	8600
com.qihoo360.mobilesafe.paysafe.payrecord.EvidenceTriggerHelper	1	24	8560	1	24	8560
com.qihoo360.mobilesafe.utils2.OptConfig	1	16	8464	1	16	8464
android.support.v4.os.ParcelableCompat	1	8	7000	1	8	7000
com.qihoo360.mobilesafe.paysafe.modules.NetProtectDataManager	1	32	4240	1	32	4240
com.qihoo.antivirus.shield.domain.ShieldDisguisePhoneMode	18	432	4080	18	432	4080
com.qihoo360.loader2.PluginProcessMain	0	0	3544	0	0	3600(+56)
com.qihoo360.mobilesafe.report.ReportConfig	0	0	3264	0	0	3264
com.qihoo360.loader2.PluginManager	1	80	2840	1	80	2840
com.qihoo360.mobilesafe.loaded.AppJar$FilterInstance	1	96	2680	1	96	2680
com.qihoo360.mobilesafe.ipcpref.IpcPrefManagerImpl	1	8	2464	1	8	2464
com.qihoo360.loader2.PluginProcessMain$ProcessPluginInfo	5	120	2208	5	120	2288(+80)
com.qihoo360.mobilesafe.block.i.BlockFactory	0	0	232	0	0	2216(+1984)
com.qihoo360.mobilesafe.loaded.client.m	0	0	2176	0	0	2176
android.support.v4.content.Loader$OnLoadCompleteListener	1	24	752	1	24	2096(+1344)

内存分析

APMCon

分析方法

meminfo → maps → smaps → mem

```
5badb000-5badc000 rw-s 00000000 b3:07 49846      /data/data/com.qihoo360.mobilesafe/shared_prefs/ipc_pref/pref_internal.shmflag
5badc000-5bae6000 r-xp 00000000 b3:07 57640      /data/data/com.qihoo360.mobilesafe/files/libmsb.so.4x.so (deleted)
5bae6000-5bae7000 rw-p 0000a000 b3:07 57640      /data/data/com.qihoo360.mobilesafe/files/libmsb.so.4x.so (deleted)
5bae7000-5bae8000 rw-p 00000000 00:00 0
5bae8000-5bae9000 ---p 00000000 00:00 0
5bae9000-5bbe8000 rw-p 00000000 00:00 0          [stack:23231]
5bbe8000-5bbf8000 rw-p 00000000 00:00 0
5bbf8000-5bbf9000 ---p 00000000 00:00 0
5bbf9000-5bcf8000 rw-p 00000000 00:00 0          [stack:23232]
5bcf8000-5bd18000 rw-p 00000000 00:00 0
5bd18000-5bd25000 r--s 007fa000 b3:07 32824      /data/app/com.qihoo360.mobilesafe-1.apk
5bd25000-5bd32000 r--s 007fa000 b3:07 32824      /data/app/com.qihoo360.mobilesafe-1.apk
```

```
          Ashmem        0        0        0
       Other dev       16       44        0
        .so mmap     1240     2736      940
       .jar mmap        4        0        0
       .apk mmap       71        0        0
       .ttf mmap        0        0        0
       .dex mmap     1652        0       24
     Other mmap     1129      336      244
        Unknown     2903      584     2896
          TOTAL    14443     9452    11436    11876     8441     3214
```

内存分析

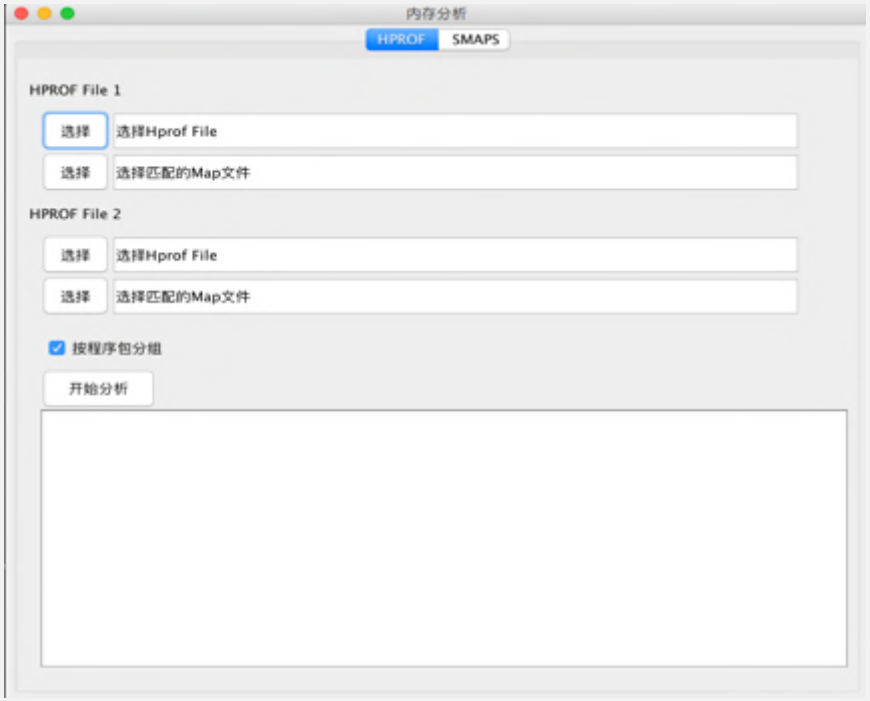
APMCon

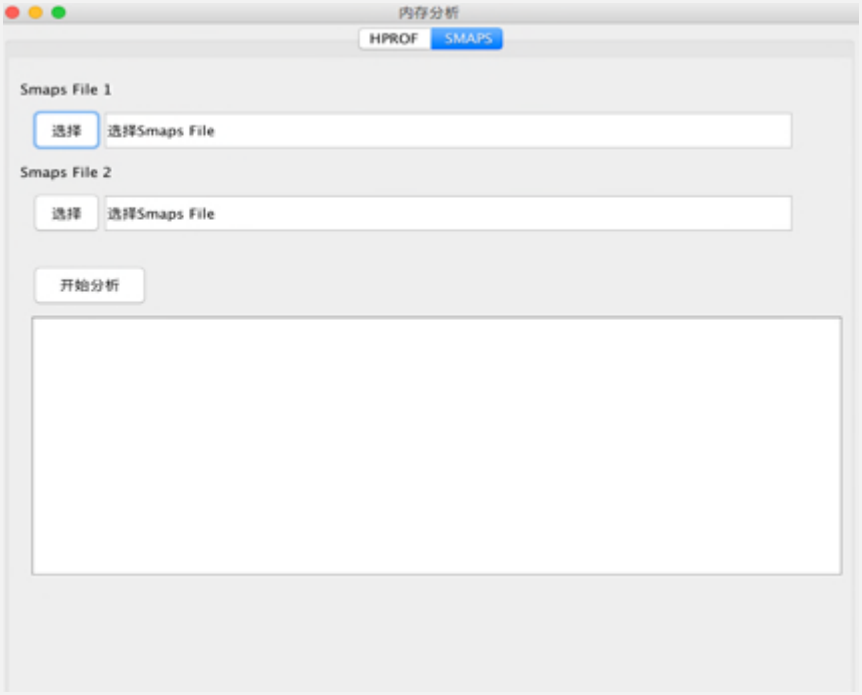
卫士 dex 名称	1058.init.smaps		1058.jiaozhun.smaps	
	PSS (kB)	RSS (kB)	PSS (kB)	RSS (kB)
/data/dalvik-cache/data@app@com.qihoo360.mobilesafe-1.apk@classes.dex	1680	3612	2400(+720)	2400(-1212)
/data/data/com.qihoo360.mobilesafe/app_plugins_v3_odex/shield-10-10-0.dex	396	396	408(+12)	408(+12)
/data/data/com.qihoo360.mobilesafe/app_plugins_v3_odex/root-10-10-0.dex	112	112	112	112
/data/dalvik-cache/data@data@com.qihoo360.mobilesafe@files@plugins_v3_data@shield@app_gabage@cache002.jar@classes.dex	33	216	35(+2)	224(+8)
/data/data/com.qihoo360.mobilesafe/files/com.qihoo360.mobilesafe_GuardService1113831752_apm.dex	32	32	32	32
/data/dalvik-cache/data@data@com.qihoo360.mobilesafe@files@plugins_v3_data@shield@app_gabage@cache001.jar@classes.dex	1	136	1	136
/data/data/com.qihoo360.mobilesafe/app_plugins_v3_odex/ntsvc-10-10-0.dex			320	320
Total	2254 percent:14.24%	4504 percent:6.01%	3308(+1054) percent:14.90%	3632(-872) percent:4.41%
卫士 so 名称	1058.init.smaps		1058.jiaozhun.smaps	
	PSS (kB)	RSS (kB)	PSS (kB)	RSS (kB)
/data/app-lib/com.qihoo360.mobilesafe-1/libmobilesafe360-jni-600.5.so	132	372	232(+100)	232(-140)
/data/app-lib/com.qihoo360.mobilesafe-1/libipc_pref.600.7.so	104	356	188(+84)	188(-168)
/data/app-lib/com.qihoo360.mobilesafe-1/libbreakpad-jni-1.3.so	40	112	64(+24)	64(-48)
/data/data/com.qihoo360.mobilesafe/files/plugins_v3_data/shield/files/libavb.so.4x.so	8	96	8	96
/data/data/com.qihoo360.mobilesafe/files/plugins_v3_data/shield/files/com.qihoo360.mobilesafe/libb.so.4x.so	4	12	4	12
Total	288 percent:1.82%	948 percent:1.27%	496(+208) percent:2.23%	592(-356) percent:0.72%
卫士 jar 名称	1058.init.smaps		1058.jiaozhun.smaps	
	PSS (kB)	RSS (kB)	PSS (kB)	RSS (kB)
/data/data/com.qihoo360.mobilesafe/app_plugins_v3/shield-10-10-0.jar	20	44	20	44
/data/data/com.qihoo360.mobilesafe/app_plugins_v3/root-10-10-0.jar	4	16	4	16
/data/data/com.qihoo360.mobilesafe/files/com.qihoo360.mobilesafe_GuardService1113831752_apm.jar	4	4	4	4
/data/data/com.qihoo360.mobilesafe/files/plugins_v3_data/shield/app_gabage/cache002.jar	0	8	0	8
/data/data/com.qihoo360.mobilesafe/files/plugins_v3_data/shield/app_gabage/cache001.jar	0	8	0	8
/data/data/com.qihoo360.mobilesafe/app_plugins_v3/ntsvc-10-10-0.jar			8	20
Total	28 percent:0.18%	80 percent:0.11%	36(+8) percent:0.16%	100(+20) percent:0.12%

内存分析

APMCon

输出工具





Argus APM

APMCon

非侵入式

支持插件性能监控

云端灵活配置

实时收集实时分析

友好的数据平台

支持针对开发人员的DEBUG模式

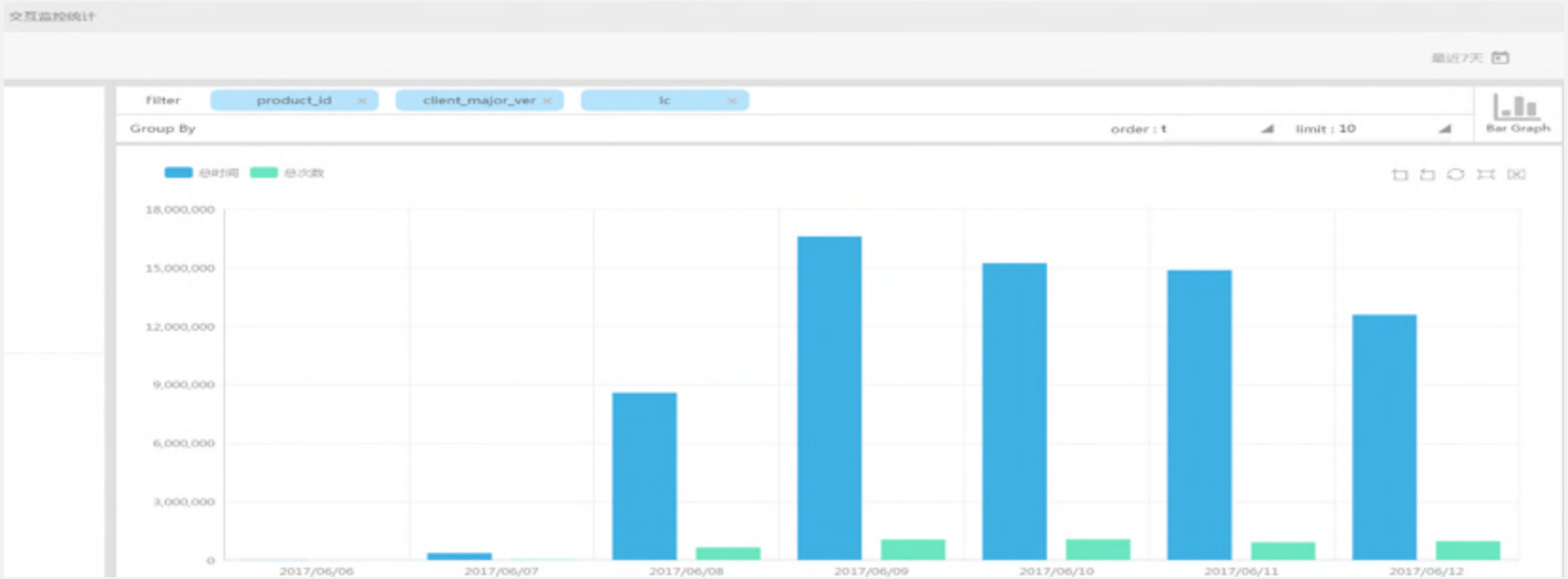
Argus APM

APMCon

- 交互体验性能
  - 应用启动
  - 组件生命周期
  - 帧率
- CPU性能
  - 瞬时使用率
  - 平均使用率
- 网络性能
  - 上传下载流量
  - 网络错误
- 电池电量
  - 电量信息
  - 耗电详情
- 内存性能
  - 进程内存信息
  - 内存详情
- IO性能
  - 文件
  - 数据库
- 传感器性能
  - 传感器信息
  - 使用时长
- ANR问题
  - ANR详情
  - ANR趋势检测

Argus APM

APMCon



实时分析

多维度过滤

实时分析

扩展能力

控制能力

漂亮的报表

Argus APM

APMCon

- DEBUG 模式
- 实时性能数据采集
- 实时本地分析
- 警告信息
- 日志记录性能问题可追溯
- 多进程支持
- 制度：接入QA Bug库

内容列表

APMCon

- 预防性能问题
- 性能监控
- 问题诊断
  - 诊断工具
  - 数据分析控制平台
- 修复技术

用户反馈

APMCon

手机诊断检测用户设备性能

- 基本信息
- 双卡信息
- ROOT信息
- 网络信息

- 自动分析耗电
- 分析屏幕密度
- 生成内存镜像文件
- Method Tracing
- CPU/内存监控
- ……

用户反馈

APMCon

手机诊断检测用户设备性能

内容列表

APMCon

- 预防性能问题
- 性能监控
- 问题诊断
- 修复技术
  - 热修复技术
  - 插件化技术

RePlugin

APMCon

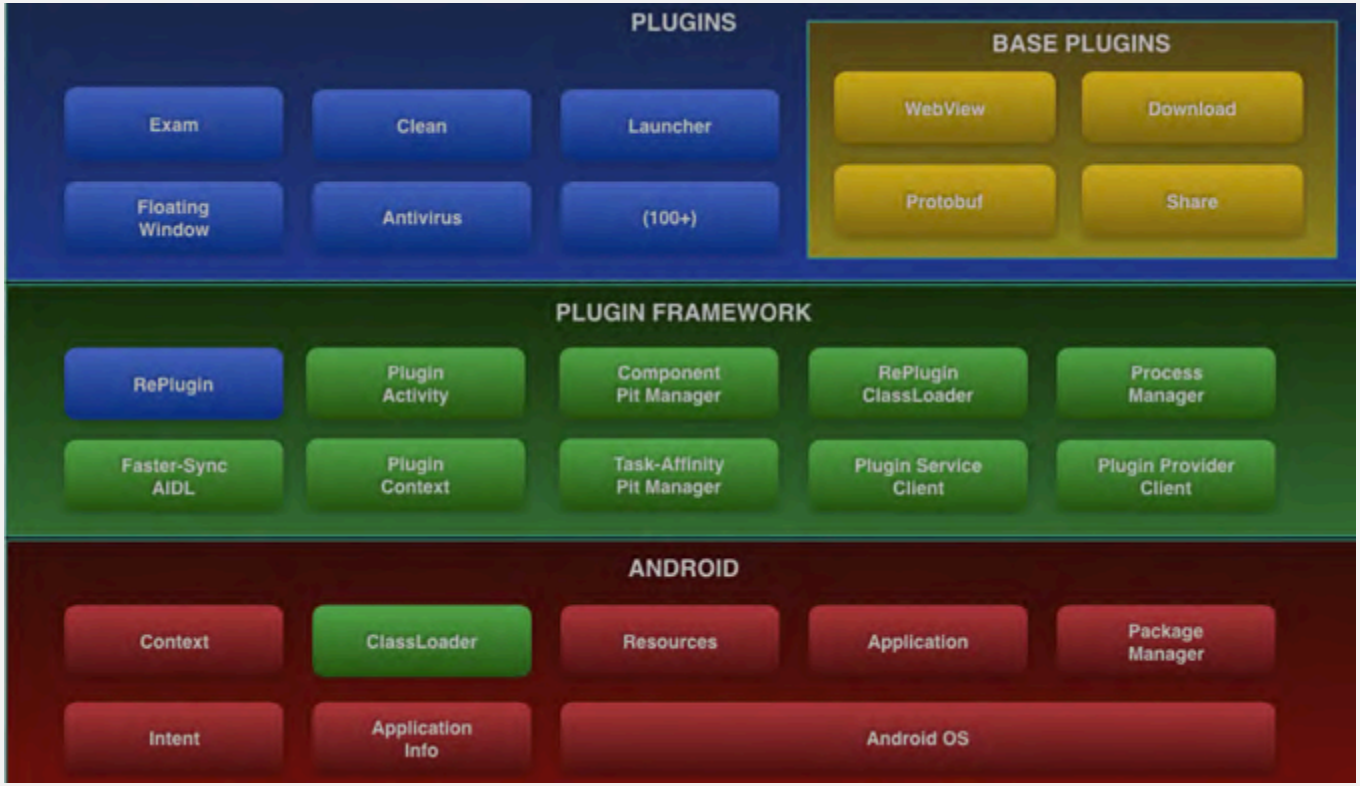
选择插件

极其灵活

非常稳定

特性丰富

易于集成



仓库地址：**https://github.com/Qihoo360/RePlugin**

总结

APMCon

手机卫士性能优化方案

- 预防性能问题
  - 代码检测工具
    - 代码火线系统
  - 自动化分析系统
  - 资源优化工具
    - 重复资源
    - 图片压缩
    - 资源混淆
    - 核心法宝：制度
  - APK构建检测
- 性能监控
  - 埋点收集信息
    - 进程冷热启动
    - 四大组件启动速度
    - 各个线程卡慢检测
    - 异步处理性能检测
  - 常规性能监控
  - 观察者工具
    - LeakCanary
    - 分析工具
    - APM集成工具——Argus APM
  - 性能分析工具
- 问题诊断
  - 诊断工具
  - 数据分析控制平台
- 修复技术
  - 热修复技术
  - 插件化技术—RePlugin

APMCon

THANK YOU

北京奇虎科技有限公司