

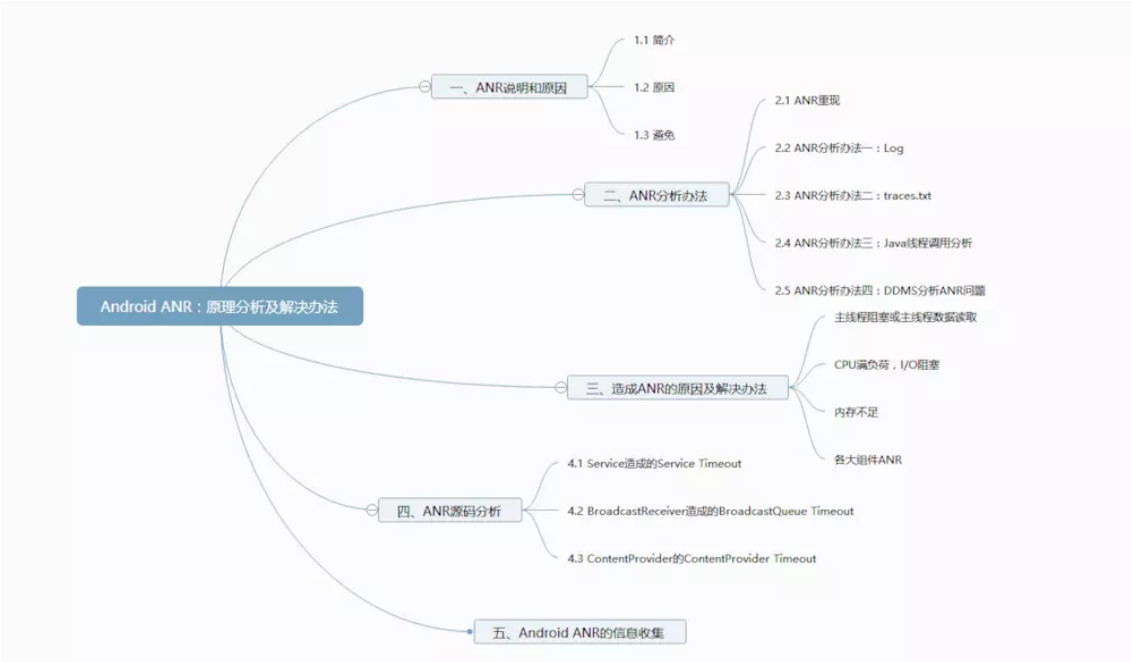
(/apps/redi
utm_sourc
banner-clic

Android ANR：原理分析及解决办法



Marker_Sky (/u/d46b4a47db84) + 关注

2017.11.09 18:05* 字数 1765 阅读 2890 评论 2 喜欢 13
(/u/d46b4a47db84)



Android ANR目录.png

一、ANR说明和原因

1.1 简介

ANR全称：**Application Not Responding**，也就是应用程序无响应。

1.2 原因

Android系统中，**ActivityManagerService(简称AMS)**和**WindowManagerService(简称WMS)**会检测App的响应时间，如果App在特定时间无法相应屏幕触摸或键盘输入时间，或者特定事件没有处理完毕，就会出现ANR。

以下四个条件都可以造成ANR发生：

- **InputDispatching Timeout**：5秒内无法响应屏幕触摸事件或键盘输入事件



- **BroadcastQueue Timeout**：在执行前台广播（BroadcastReceiver）的 onReceive() 函数时10秒没有处理完成，后台为60秒。
- **Service Timeout**：前台服务20秒内，后台服务在200秒内没有执行完毕。
- **ContentProvider Timeout**：ContentProvider的publish在10s内没进行完。

(/apps/redi
utm_sourc
banner-cl

1.3 避免

尽量避免在主线程（UI线程）中作耗时操作。

那么耗时操作就放在子线程中。

关于多线程可以参考：Android多线程：理解和简单使用总结

(<https://www.jianshu.com/p/56163a3beb4a>)

二、ANR分析办法

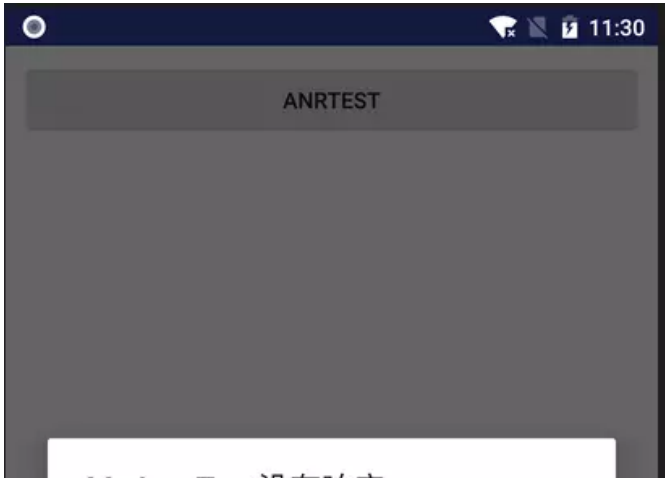
2.1 ANR重现

这里使用的是号称Google亲儿子的Google Pixel xl（Android 8.0系统）做的测试，生成一个按钮跳转到 ANRTestActivity，在后者的 onCreate() 中主线程休眠20秒：

```
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_anr_test);
    // 这是Android提供线程休眠函数，与Thread.sleep()最大的区别是
    // 该使用该函数不会抛出InterruptedException异常。
    SystemClock.sleep(20 * 1000);
}
```

在进入 ANRTestActivity 后黑屏一段时间，大概有七八秒，终于弹出了ANR异常。





(/apps/redi
utm_sourc
banner-clic

Google Pixel xl ANR.png

2.2 ANR分析办法一：Log

刚才产生ANR后，看下Log：

```
11-07 11:33:57.554 23346-23364/com.sky.myapplication I/zygote64: Thread[3,tid=23364,WaitingInMainSignalCatcherLoop,Thread*=0x74a4bf400,peer=0x13340020,"Signal Catcher"]: reacting to signal 3
11-07 11:33:57.594 23346-23364/com.sky.myapplication I/zygote64: Wrote stack traces to '/data/anr/traces.txt'
11-07 11:34:11.283 23346-23346/com.sky.myapplication I/Choreographer: Skipped 1208 frames! The application may be doing too much work on its main thread.
```

ANR Log.png

可以看到logcat清晰地记录了ANR发生的时间，以及线程的tid和一句话概括原因：
WaitingInMainSignalCatcherLoop，大概意思为主线程等待异常。
最后一句 The application may be doing too much work on its main thread. 告知可能在主线程做了太多的工作。

2.3 ANR分析办法二：traces.txt

刚才的log有第二句 Wrote stack traces to '/data/anr/traces.txt'，说明ANR异常已经输出到 traces.txt 文件，使用adb命令把这个文件从手机里导出来：

- 1. cd到 adb.exe 所在的目录，也就是**Android SDK**的 platform-tools 目录，例如：

```
cd D:\Android\AndroidSdk\platform-tools
```

此外，除了**Windows**的cmd以外，还可以使用**AndroidStudio**的**Terminal**来输入adb命令。

- 2. 到指定目录后执行以下adb命令导出 traces.txt 文件：



```
adb pull /data/anr/traces.txt
```

traces.txt 默认会被导出到**Android SDK**的\platform-tools目录。一般来说
traces.txt 文件记录的东西会比较多，分析的时候需要有针对性地去查找相关记录。

(/apps/redi
utm_sourc
banner-clic

```
----- pid 23346 at 2017-11-07 11:33:57 ----- -----> 进程id和ANR产生时间
Cmd line: com.sky.myjavatest
Build fingerprint: 'google/marlin/marlin:8.0.0/OPR3.170623.007/4286350:u
ABI: 'arm64'
Build type: optimized
Zygote loaded classes=4681 post zygote classes=106
Intern table: 42675 strong; 137 weak
JNI: CheckJNI is on; globals=526 (plus 22 weak)
Libraries: /system/lib64/libandroid.so /system/lib64/libcompiler_rt.so
/system/lib64/libjavacrypto.so
/system/lib64/libjnigraphics.so /system/lib64/libmedia_jni.so /system/li
/system/lib64/libwebviewchromium_loader.so libjavacore.so libopenjdk.so
Heap: 22% free, 1478KB/1896KB; 21881 objects -----> 内存使用情况

...

"main" prio=5 tid=1 Sleeping -----> 原因为Sleeping
  | group="main" sCount=1 dsCount=0 flags=1 obj=0x733d0670 self=0x74a4ab
  | sysTid=23346 nice=-10 cgrp=default sched=0/0 handle=0x74a91ab9b0
  | state=S schedstat=( 391462128 82838177 354 ) utm=33 stm=4 core=3 HZ=
  | stack=0x7fe6fac000-0x7fe6fae000 stackSize=8MB
  | held mutexes=
  at java.lang.Thread.sleep(Native method)
  - sleeping on <0x053fd2c2> (a java.lang.Object)
  at java.lang.Thread.sleep(Thread.java:373)
  - locked <0x053fd2c2> (a java.lang.Object)
  at java.lang.Thread.sleep(Thread.java:314)
  at android.os.SystemClock.sleep(SystemClock.java:122)
  at com.sky.myjavatest.ANRTestActivity.onCreate(ANRTestActivity.java:20
  at android.app.Activity.performCreate(Activity.java:6975)
  at android.app.Instrumentation.callActivityOnCreate(Instrumentation.ja
  at android.app.ActivityThread.performLaunchActivity(ActivityThread.jav
  at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java
  at android.app.ActivityThread.-wrap11(ActivityThread.java:-1)
  at android.app.ActivityThread$H.handleMessage(ActivityThread.java:1593
  at android.os.Handler.dispatchMessage(Handler.java:105)
  at android.os.Looper.loop(Looper.java:164)
  at android.app.ActivityThread.main(ActivityThread.java:6541)
  at java.lang.reflect.Method.invoke(Native method)
  at com.android.internal.os.Zygote$MethodAndArgsCaller.run(Zygote.java:
  at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:767)
```

在文件中使用 **ctrl + F** 查找包名可以快速定位相关代码。

通过上方log可以看出相关问题：

- 进程id和包名： pid 23346 com.sky.myjavatest
- 造成ANR的原因： Sleeping
- 造成ANR的具体行数： ANRTestActivity.java:20 类的第20行



特别注意：产生新的**ANR**，原来的 **traces.txt** 文件会被覆盖。

2.4 ANR分析办法三：Java线程调用分析

通过JDK提供的命令可以帮助分析和调试Java应用，命令为：

```
jstack {pid}
```

其中pid可以通过jps命令获得，jps命令会列出当前系统中运行的所有Java虚拟机进程，比如

```
7266 Test
7267 Jps
```

具体分析参考：Android应用ANR分析
(<https://www.jianshu.com/p/30c1a5ad63a3>) 四.1节

2.5 ANR分析办法四：DDMS分析ANR问题

- 使用DDMS——Update Threads工具
- 阅读Update Threads的输出

具体分析参考：Android应用ANR分析
(<https://www.jianshu.com/p/30c1a5ad63a3>) 四.2节

三、造成ANR的原因及解决办法

上面例子只是由于简单的主线程耗时操作造成的ANR，造成ANR的原因还有很多：

- 主线程阻塞或主线程数据读取

解决办法：避免死锁的出现，使用子线程来处理耗时操作或阻塞任务。尽量避免在主线程query provider、不要滥用SharedPreferences (<https://link.jianshu.com?t=http://weishu.me/2016/10/13/sharedpreference-advice/>)

- CPU满负荷，I/O阻塞

解决办法：文件读写或数据库操作放在子线程异步操作。

- 内存不足

(/apps/redi
utm_sourc
banner-clic



解决办法: AndroidManifest.xml 文件<application>中可以设置 android:largeHeap="true", 以此增大App使用内存。不过**不建议使用此法**, 从根本上防止内存泄漏, 优化内存使用才是正道。

(/apps/redi
utm_sourc
banner-clic

- 各大组件ANR

各大组件生命周期中也应避免耗时操作, 注意BroadcastReceiver的 onReceive()、后台Service和ContentProvider也不要执行太长时间的任务。

四、ANR源码分析

特别声明: 文章 理解Android ANR的触发原理 (<https://link.jianshu.com?t=http://gityuan.com/2016/07/02/android-anr/>) 分别记录了由**Service**、**BroadcastReceiver**和**ContentProvider**造成的ANR。下文引用该文代码, 并依据自己的简单理解作总结。

4.1 Service造成的Service Timeout

Service Timeout是位于"**ActivityManager**"线程中的AMS.MainHandler收到 SERVICE_TIMEOUT_MSG 消息时触发。

4.1.1 发送延时消息

Service进程attach到system_server进程的过程中会调用 realStartServiceLocked, 紧接着 mAm.mHandler.sendMessageAtTime() 来发送一个延时消息, 延时的时常是定义好的, 如前台**Service**的20秒。**ActivityManager**线程中的**AMS.MainHandler**收到 SERVICE_TIMEOUT_MSG 消息时会触发。

AS.realStartServiceLocked

ActiveServices.java



```

private final void realStartServiceLocked(ServiceRecord r,
    ProcessRecord app, boolean execInFg) throws RemoteException {
    ...
    //发送delay消息(SERVICE_TIMEOUT_MSG)
    bumpServiceExecutingLocked(r, execInFg, "create");
    try {
        ...
        //最终执行服务的onCreate()方法
        app.thread.scheduleCreateService(r, r.serviceInfo,
            mAm.compatibilityInfoForPackageLocked(r.serviceInfo.appPackage,
                app.repProcState);
    } catch (DeadObjectException e) {
        mAm.appDiedLocked(app);
        throw e;
    } finally {
        ...
    }
}

```

(/apps/redi
utm_sourc
banner-clic

AS.bumpServiceExecutingLocked

```

private final void bumpServiceExecutingLocked(ServiceRecord r, boolean f
    ...
    scheduleServiceTimeoutLocked(r.app);
}

void scheduleServiceTimeoutLocked(ProcessRecord proc) {
    if (proc.executingServices.size() == 0 || proc.thread == null) {
        return;
    }
    long now = SystemClock.uptimeMillis();
    Message msg = mAm.mHandler.obtainMessage(
        ActivityManagerService.SERVICE_TIMEOUT_MSG);
    msg.obj = proc;

    //当超时后仍没有remove该SERVICE_TIMEOUT_MSG消息, 则执行service Timeout流程
    mAm.mHandler.sendMessageAtTime(msg,
        proc.execServicesFg ? (now+SERVICE_TIMEOUT) : (now+ SERVICE_BACK
}

```

4.1.2 进入目标进程的主线程创建Service

经过Binder等层层调用进入目标进程的主线程

handleCreateService(CreateServiceData data)。

ActivityThread.java



```

private void handleCreateService(CreateServiceData data) {
    ...
    java.lang.ClassLoader cl = packageInfo.getClassLoader();
    Service service = (Service) cl.loadClass(data.info.name).newInstance
    ...

    try {
        //创建ContextImpl对象
        ContextImpl context = ContextImpl.createAppContext(this, package
        context.setOuterContext(service);
        //创建Application对象
        Application app = packageInfo.makeApplication(false, mInstrument
        service.attach(context, this, data.info.name, data.token, app,
            ActivityManagerNative.getDefault());
        //调用服务onCreate()方法
        service.onCreate();

        //取消AMS.MainHandler的延时消息
        ActivityManagerNative.getDefault().serviceDoneExecuting(
            data.token, SERVICE_DONE_EXECUTING_ANON, 0, 0);
    } catch (Exception e) {
        ...
    }
}

```

(/apps/redi
utm_sourc
banner-clic

这个方法中会创建目标服务对象，以及回调常用的**Service**的 `onCreate()` 方法，紧接着通过 `serviceDoneExecuting()` 回到 `system_server` 执行取消 `AMS.MainHandler` 的延时消息。

4.1.3 回到 `system_server` 执行取消 `AMS.MainHandler` 的延时消息

AS.serviceDoneExecutingLocked

```

private void serviceDoneExecutingLocked(ServiceRecord r, boolean inDestro
    boolean finishing) {
    ...
    if (r.executeNesting <= 0) {
        if (r.app != null) {
            r.app.execServicesFg = false;
            r.app.executingServices.remove(r);
            if (r.app.executingServices.size() == 0) {
                //当前服务所在进程中没有正在执行的service
                mAm.mHandler.removeMessages(ActivityManagerService.SERVI
            ...
        }
        ...
    }
}

```

此方法中 `Service` 逻辑处理完成则移除之前延时的消息 `SERVICE_TIMEOUT_MSG`。如果没有执行完毕不调用这个方法，则超时后会发出 `SERVICE_TIMEOUT_MSG` 来告知 ANR 发生。

4.2 BroadcastReceiver造成的BroadcastQueue Timeout



BroadcastReceiver Timeout是位于"ActivityManager"线程中的
BroadcastQueue.BroadcastHandler收到 BROADCAST_TIMEOUT_MSG 消息时触发。

(/apps/redi
utm_sourc
banner-clic

4.2.1 处理广播函数 processNextBroadcast() 中 broadcastTimeoutLocked(false) 发送延时消息

广播处理顺序为先处理并行广播，再处理当前有序广播。

```
final void processNextBroadcast(boolean fromMsg) {
    synchronized(mService) {
        ...
        // 处理当前有序广播
        do {
            r = mOrderedBroadcasts.get(0);
            //获取所有该广播所有的接收者
            int numReceivers = (r.receivers != null) ? r.receivers.size() : 0;
            if (mService.mProcessesReady && r.dispatchTime > 0) {
                long now = SystemClock.uptimeMillis();
                if ((numReceivers > 0) &&
                    (now > r.dispatchTime + (2*mTimeoutPeriod*numReceivers))) {
                    //step 1. 发送延时消息，这个函数处理了很多事情，比如广播处理
                    broadcastTimeoutLocked(false);
                    ...
                }
            }
            if (r.receivers == null || r.nextReceiver >= numReceivers
                || r.resultAbort || forceReceive) {
                if (r.resultTo != null) {
                    //2. 处理广播消息消息
                    performReceiveLocked(r.callerApp, r.resultTo,
                        new Intent(r.intent), r.resultCode,
                        r.resultData, r.resultExtras, false, false, r.userLabel);
                    r.resultTo = null;
                }
                //3. 取消广播超时ANR消息
                cancelBroadcastTimeoutLocked();
            }
        } while (r == null);
        ...

        // 获取下条有序广播
        r.receiverTime = SystemClock.uptimeMillis();
        if (!mPendingBroadcastTimeoutMessage) {
            long timeoutTime = r.receiverTime + mTimeoutPeriod;
            //设置广播超时
            setBroadcastTimeoutLocked(timeoutTime);
        }
        ...
    }
}
```

上文的step 1. broadcastTimeoutLocked(false)函数：记录时间信息并调用函数设置发送延时消息



```

final void broadcastTimeoutLocked(boolean fromMsg) {
    ...
    long now = SystemClock.uptimeMillis();
    if (fromMsg) {
        if (mService.mDidDexOpt) {
            // Delay timeouts until dexopt finishes.
            mService.mDidDexOpt = false;
            long timeoutTime = SystemClock.uptimeMillis() + mTimeoutPeriod;
            setBroadcastTimeoutLocked(timeoutTime);
            return;
        }
        if (!mService.mProcessesReady) {
            return;
        }

        long timeoutTime = r.receiverTime + mTimeoutPeriod;
        if (timeoutTime > now) {
            // step 2
            setBroadcastTimeoutLocked(timeoutTime);
            return;
        }
    }
}

```

(/apps/redi
utm_sourc
banner-clic

上文的**step 2.setBroadcastTimeoutLocked**函数：设置广播超时具体操作，同样是发送延时消息

```

final void setBroadcastTimeoutLocked(long timeoutTime) {
    if (!mPendingBroadcastTimeoutMessage) {
        Message msg = mHandler.obtainMessage(BROADCAST_TIMEOUT_MSG, this);
        mHandler.sendMessageAtTime(msg, timeoutTime);
        mPendingBroadcastTimeoutMessage = true;
    }
}

```

4.2.2 setBroadcastTimeoutLocked(long timeoutTime)函数的参数
timeoutTime是当前时间加上设定好的超时时间。

也就是上文的

```

long timeoutTime = SystemClock.uptimeMillis() + mTimeoutPeriod;

```

mTimeoutPeriod 也就是前台队列的10s和后台队列的60s。



```

public ActivityManagerService(Context systemContext) {
    ...
    static final int BROADCAST_FG_TIMEOUT = 10 * 1000;
    static final int BROADCAST_BG_TIMEOUT = 60 * 1000;
    ...
    mFgBroadcastQueue = new BroadcastQueue(this, mHandler,
        "foreground", BROADCAST_FG_TIMEOUT, false);
    mBgBroadcastQueue = new BroadcastQueue(this, mHandler,
        "background", BROADCAST_BG_TIMEOUT, true);
    ...
}

```

(/apps/redi
utm_sourc
banner-clic

4.2.3 在processNextBroadcast()过程，执行完performReceiveLocked后调用cancelBroadcastTimeoutLocked

cancelBroadcastTimeoutLocked：处理广播消息函数 processNextBroadcast() 中 performReceiveLocked() 处理广播消息完毕则调用 cancelBroadcastTimeoutLocked() 取消超时消息。

```

final void cancelBroadcastTimeoutLocked() {
    if (mPendingBroadcastTimeoutMessage) {
        mHandler.removeMessages(BROADCAST_TIMEOUT_MSG, this);
        mPendingBroadcastTimeoutMessage = false;
    }
}

```

4.3 ContentProvider的ContentProvider Timeout

ContentProvider Timeout是位于"ActivityManager"线程中的AMS.MainHandler收到CONTENT_PROVIDER_PUBLISH_TIMEOUT_MSG消息时触发。

参考理解Android ANR的触发原理 (<https://link.jianshu.com?t=http://gityuan.com/2016/07/02/android-anr/>)第四节

五、Android ANR的信息收集

无论是四大组件或者进程等只要发生ANR，最终都会调用AMS.appNotResponding()方法。

参考：理解Android ANR的信息收集过程 (<https://link.jianshu.com?t=http://gityuan.com/2016/12/02/app-not-response/>)

参考资料：




理解Android ANR的触发原理 (<https://link.jianshu.com?t=http://gityuan.com/2016/07/02/android-anr/>)
理解Android ANR的信息收集过程 (<https://link.jianshu.com?t=http://gityuan.com/2016/12/02/app-not-response/>)
Android App优化之ANR详解 (<https://www.jianshu.com/p/6d855e984b99>)
Android 源码分析ANR (<https://link.jianshu.com?t=http://blog.csdn.net/jasonwang18/article/details/60326807>)

(/apps/redi
utm_sourc
banner-clic

小礼物走一走，来简书关注我

赞赏支持

 Android 内存与性能优化 (/nb/18543719)

举报文章 © 著作权归作者所有



Marker_Sky (/u/d46b4a47db84) ♂

写了 74545 字，被 87 人关注，获得了 178 个喜欢
(/u/d46b4a47db84)

+ 关注

可以虚怀若谷 不可妄自菲薄 谢谢大家的小心心♥

喜欢 | 13



更多分享

开发10年
全记在这本Java进阶宝典了

Spring源码分析

分布式架构

微服务架构

JVM性能优化

高效DevOps

多线程并发编程

点击领取

(/p/428251ede1aa)



登录 (/sign-in?source=desktop&utm_medium=not-si


发表评论




-comr


2条评论 只看作者

按时间倒序 按时间正序



AWeiLoveAndroid (/u/f408bdadacce) 

2楼 · 2018.05.01 08:02

(/u/f408bdadacce)
手机存储不足100M 使用QQ会造成ANR 清理内存后 再打开QQ偶尔有卡顿 我经常遇到




赞 回复

Marker_Sky (/u/d46b4a47db84): @AWeiLoveAndroid (/users/f408bdadacce) QQ 现在占的内存能到 1G 多，前些年想都不敢想。
2018.05.01 19:29 回复

添加新评论


(/apps/redi
utm_sourc
banner-clic

被以下专题收入，发现更多相似内容

-  Android... (/c/8f0b97c68ff1?utm_source=desktop&utm_medium=notes-included-collection)
-  android... (/c/4f001e9de29d?utm_source=desktop&utm_medium=notes-included-collection)


掘金 Android 文章精选合集 (/p/5ad013eb5364?utm_campaign=maleski...

用两张图告诉你，为什么你的 App 会卡顿？ - Android - 掘金 Cover 有什么料？ 从这篇文章中你能获得这些料： 知道setContentView()之后发生了什么？ ... Android 获取 View 宽高的常用正确方式， ...

 掘金官方 (/u/5fc9b6410f4f?
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio


Android - 收藏集 (/p/dad51f6c9c4d?utm_campaign=maleskine&utm_c...

用两张图告诉你，为什么你的 App 会卡顿？ - Android - 掘金 Cover 有什么料？ 从这篇文章中你能获得这些料： 知道setContentView()之后发生了什么？ ... Android 获取 View 宽高的常用正确方式， ...

 passiontim (/u/e946d18f163c?
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio

给解决问题ANR一个印象 (/p/90eede51df55?utm_campaign=maleskine&...

简介 现在感觉自己做的工作，基本上脱离Android了就是用java写代码，而且可能试用期完了就会被刷，很多东西是公司自己的，完全不知道怎么下手研究，导师指导也就是几句话的说你看下哪里哪...

 我叫王菜鸟 (/u/6162cec1158d?
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=rec



(/p/3dcdd12e5a5d?




(/apps/redi

utm_sourc
banner-clic

utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recom
Android ANR分析 (/p/3dcdd12e5a5d?utm_campaign=maleskine&utm...

什么是ANR ANR(Application Not Responding)就是应用在规定的时间内没有响应用户输入或者其他应用或者系统服务。发生ANR的场景 Service超时 Service ANR一般是指AMS通过binder IPC调用...


 lbtrace (/u/742d7f638281?

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio

如何分析解决Android ANR (/p/f4543a729619?utm_campaign=maleskin...

===== 一：什么是

ANRANR:Application Not Responding, 即应用无响应 二： ANR的类型ANR一般有三种类型： 1： ...

 爱情小傻蛋 (/u/897ea90d2e83?


utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio

(/p/63519812d2ad?



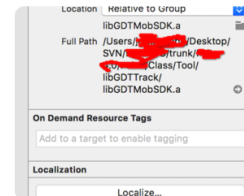
utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendatio
百年乌酱顺德辣菜 (/p/63519812d2ad?utm_campaign=maleskine&utm_c...

顺德百丈园 在顺德有一味传统的酱料，特别适合潮湿多雨的广东，那就是祛湿暖身的乌酱，用黄豆煮水加蒜蓉、当地的辣椒发酵而成，这便是顺德龙江龙山当地村民家家户户都会做的酱料，既颠覆了...

 吃货小队 (/u/b31bd5d94a8f?


utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio

(/p/a7a40f5d0d27?



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendatio
linker command failed with exit code 1 (use -v ... (/p/a7a40f5d0d27?ut...

更新第三方库,出现了这个问题,怎么办?我看到好多帖子说应该把库删了再加,或者别的什么的.我的方法可能是最快的方式吧.找到对应的库,点开它的.m或者是.a文件,看看他的Target Membership 有没...

 十一妖 (/u/a79a9392d831?


utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio



一起光头 (/p/bca1abbd0eb5?utm_campaign=maleskine&utm_content=...

《滚蛋吧，肿瘤君》两小时的电影，却是熊顿最后的时光。疾病和明天，谁也不知道谁会先来到，所


以，珍惜当下！但是，疾病和明天，那么理所当然的一定会出现，你又该如何招架。熊顿的个性使...

 ab15b9e2370f (/u/ab15b9e2370f?
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio

 (/apps/redi
utm_sourc
banner-clic

中秋快乐。 (/p/74a88ca4d885?utm_campaign=maleskine&utm_content=...

慵懒在床上。不知到底是怎么样。我想不通。也走不出自己的那个圈。我想好好的。我不想再这个样
子下去。或许。我。需要有人拉我一把。讨厌社交。不理。不睬。只是想知道，当我低入谷底。甚...

 AEAT (/u/d620bdc0b27c?
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendatio

