Anr 分析指导文档

1）什么引发了 ANR?

在 Android 里，应用程序的响应性是由 Activity Manager 和 WindowManager 系统服务监视的。当它监测到以下情况中的一个时，Android 就会针对特定的应用程序显示 ANR：

在 5 秒内没有响应输入的事件（例如，按键按下，屏幕触摸）
BroadcastReceiver 在 10 秒内没有执行完毕

一个 ANR 对话框显示给用户

2）如何避免 ANR?

考虑上面的 ANR 定义，让我们来研究一下为什么它会在 Android 应用程序里发生和如何最佳构建应用程序来避免 ANR。

Android 应用程序通常是运行在一个单独的线程（例如，main）里。这意味着你的应用程序所做的事情如果在主线程里占用了太长的时间的话，就会引发 ANR 对话框，因为你的应用程序并没有给自己机会来处理输入事件或者 Intent 广播。

因此，运行在主线程里的任何方法都尽可能少做事情。特别是，Activity 应该在它的关键生命周期方法（如 onCreate()和 onResume()）里尽可能少的去做创建操作。潜在的耗时操作，例如网络或数据库操作，或者高耗时的计算如改变位图尺寸，应该在子线程里（或者以数据 库操作为例，通过异步请求的方式）来完成。然而，不是说你的主线程阻塞在那里等待子线程的完成——也不是调用 Thread.wait()或是 Thread.sleep()。替代的方法是，主线程应该为子线程提供一个 Handler，以便完成时能够提交给主线程。以这种方式设计你的应用程序，将 能保证你的主线程保持对输入的响应性并能避免由于 5 秒输入事件的超时引发的 ANR 对话框。这种做法应该在其它显示 UI 的线程里效仿，因为它们都受相同的超 时影响。

IntentReceiver 执行时间的特殊限制意味着它应该做：在后台里做小的、琐碎的工作如保存设定或者注册一个 Notification。和在主线 程里调用的其它方法一样，应用程序应该避免在 BroadcastReceiver 里做耗时的操作或计算。但不再是在子线程里做这些任务（因为 BroadcastReceiver 的生命周期短），替代的是，如果响应 Intent 广播需要执行一个耗时的动作的话，应用程序应该启动一个 Service。顺便提及一句，你也应该避免在 IntentReceiver 里启动一个 Activity，因为它会创建一个新的画面，并从当前用户正在运行的程序上抢夺焦点。如果你的应用程序在响应 Intent 广播时需要向用户展示什么，你应该使用 Notification Manager 来实现。

3）增强响应灵敏性

一般来说，在应用程序里，100 到 200ms 是用户能感知阻滞的时间阈值。因此，这里有一些额外的技巧来避免 ANR，并有助于让你的应用程序看起来有响应性。

如果你的应用程序为响应用户输入正在后台工作的话，可以显示工作的进度（ProgressBar 和 ProgressDialog 对这种情况来说很有用）。
特别是游戏，在子线程里做移动的计算。
如果你的应用程序有一个耗时的初始化过程的话，考虑可以显示一个 SplashScreen 或者快速显示主画面并异步来填充这些信息。在这两种情况下，你都应该显示正在进行的进度，以免用户认为应用程序被冻结了。

15:59:37 I/ActivityManager(130): ANR in process: com.android.email(last in com.android.email)

=>frameworks\base\services\java\com\android\server\am\ActivityManagerService.java
=>提示输出 cpu 信息
Annotation: keyDispatchingTimedOut
CPU usage:

=>frameworks\base\services\java\com\android\server\ProcessStats.java
=>输出 cpu 当前状态
=>/proc/loadavg 显示 cpu 负荷
=>1-分钟平均负载 / 5-分钟平均负载 / 15-分钟平均负载
Load: 4.37 /4.55 / 3.97
=>cpu 状态的时间段
CPU usage from 10987ms to 27ms ago:
=>/proc/state 读取 cpu 的使用情况
=>http://linux.die.net/man/5/proc
=>user
=>kernel
=>iowait
=>irq ->0
=>softirq ->0
=>minor
The number of minor faults the process has madewhich have not required loading a memory page fromdisk.
=>major
The number of major faults the process has madewhich have required loading a memory page from disk.
system_server: 12% = 4% user + 7% kernel /faults: 1886 minor
m.android.email: 12% = 6% user + 5% kernel /faults: 2716 minor

```
  sensorserver_ya: 7% = 0% user + 7%kernel
  breeze.launcher: 3% = 0% user + 3% kernel /faults: 94 minor
  ocess.msn.shell: 0% = 0% user + 0% kernel /faults: 38 minor
  m.android.phone: 0% = 0% user + 0%kernel
  alog: 0% = 0% user + 0% kernel
  rpcrotuer_smd_x: 0% = 0% user + 0%kernel
  rild: 0% = 0% user + 0% kernel
  alog: 0% = 0% user + 0% kernel
  events/0: 0% = 0% user + 0% kernel
  port-bridge: 0% = 0% user + 0% kernel
TOTAL: 81% = 13% user + 25% kernel + 42% iowait

15:59:37 I/ActivityManager(130): Removing old ANR trace file
from/data/anr/traces.txt
```

Android ANR 这个错误大家并不陌生，但是从 Android2.2 开始出错的 ANR 信息会自动上传给 Google 进行系统分析改进，当然了你的应用 ANR 错误其实保存在一个文件中，在 /data/anr/traces.txt 文件中.
下面一起来分析下错误吧,第一行为出错的时间,第二行都会写上发生 ANR 的 packageName，下文是 com.android.systemui 这个 包，里面的部分线程出了问题，通过下面的的 xxx 方法以及对应的 java 文件，后面的数字为 xxx.java 文件的第几行，是不是很方便呢?

```
----- pid 125 at 2011-02-22 05:18:01 -----
Cmd line: com.android.systemui

DALVIK THREADS:
(mutexes: tll=0 tsl=0 tscl=0 ghl=0 hwl=0 hwll=0)
"main" prio=5 tid=1 NATIVE
  | group="main" sCount=1 dsCount=0 obj=0x4001f1a8self=0xce48
  | sysTid=125 nice=0 sched=0/0 cgrp=defaulthandle=-1345006528
  | schedstat=( 981213067 8042604425 151 )
  at android.os.BinderProxy.transact(NativeMethod)
  atandroid.os.storage.IMountService$Stub$Proxy.isUsbMassStorageConnected(IMount
tService.java:95)
  atandroid.os.storage.StorageManager.isUsbMassStorageConnected(StorageManager.
java:385)
  atcom.android.systemui.usb.StorageNotification.<init>(StorageNotification.jav
a:71)
  atcom.android.systemui.statusbar.policy.StatusBarPolicy.<init>(StatusBarPolic
y.java:412)
  atcom.android.systemui.statusbar.StatusBarService.onCreate(StatusBarService.j
ava:239)
  atandroid.app.ActivityThread.handleCreateService(ActivityThread.java:1920)
  atandroid.app.ActivityThread.access$2500(ActivityThread.java:117)
  atandroid.app.ActivityThread$H.handleMessage(ActivityThread.java:982)
```

```
  atandroid.os.Handler.dispatchMessage(Handler.java:99)
  at android.os.Looper.loop(Looper.java:123)
  atandroid.app.ActivityThread.main(ActivityThread.java:3647)
  at java.lang.reflect.Method.invokeNative(NativeMethod)
  atjava.lang.reflect.Method.invoke(Method.java:507)
  atcom.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:
839)
  atcom.android.internal.os.ZygoteInit.main(ZygoteInit.java:597)
  at dalvik.system.NativeStart.main(NativeMethod)

"Binder Thread #2" prio=5 tid=8 NATIVE
  | group="main" sCount=1 dsCount=0 obj=0x40511270self=0x1c1100
  | sysTid=153 nice=0 sched=0/0 cgrp=defaulthandle=1141608
  | schedstat=( 27181321 64708345 5 )
  at dalvik.system.NativeStart.run(NativeMethod)

"Binder Thread #1" prio=5 tid=7 NATIVE
  | group="main" sCount=1 dsCount=0 obj=0x405111a8self=0x1349c8
  | sysTid=152 nice=0 sched=0/0 cgrp=defaulthandle=1264008
  | schedstat=( 31857510 74284459 5 )
  at dalvik.system.NativeStart.run(NativeMethod)

"Compiler" daemon prio=5 tid=6 VMWAIT
  | group="system" sCount=1 dsCount=0obj=0x4050dd10 self=0x116538
  | sysTid=144 nice=0 sched=0/0 cgrp=defaulthandle=982520
  | schedstat=( 7319346 103454345 12 )
  at dalvik.system.NativeStart.run(NativeMethod)

"JDWP" daemon prio=5 tid=5 VMWAIT
  | group="system" sCount=1 dsCount=0obj=0x4050dc60 self=0x116400
  | sysTid=142 nice=0 sched=0/0 cgrp=defaulthandle=986536
  | schedstat=( 32876840 393298925 20 )
  at dalvik.system.NativeStart.run(NativeMethod)

"Signal Catcher" daemon prio=5 tid=4 RUNNABLE
  | group="system" sCount=0 dsCount=0obj=0x4050dba0 self=0x253ab0
  | sysTid=141 nice=0 sched=0/0 cgrp=defaulthandle=2439792
  | schedstat=( 94447996 796994478 19 )
  at dalvik.system.NativeStart.run(NativeMethod)

"GC" daemon prio=5 tid=3 VMWAIT
  | group="system" sCount=1 dsCount=0obj=0x4050daf8 self=0x8fd40
  | sysTid=128 nice=0 sched=0/0 cgrp=defaulthandle=2439592
  | schedstat=( 103352102 819201761 21 )
  at dalvik.system.NativeStart.run(NativeMethod)
```

```
"HeapWorker" daemon prio=5 tid=2 VMWAIT
  | group="system" sCount=1 dsCount=0obj=0x4050da40 self=0xf0c70
  | sysTid=127 nice=0 sched=0/0 cgrp=defaulthandle=2439528
  | schedstat=( 971134410 6445300652 103 )
  at dalvik.system.NativeStart.run(NativeMethod)
```

二：ANR 的常见类型

ANR 一般有三种类型：

1：**KeyDispatchTimeout(5 seconds) --主要类型**

按键或触摸事件在特定时间内无响应

2：BroadcastTimeout(10 seconds)

BroadcastReceiver 在特定时间内无法处理完成

3：**ServiceTimeout(20 seconds) --小概率类型**

Service 在特定的时间内无法处理完成

三：**KeyDispatchTimeout**

Akey or touch event was not dispatched within the specified time（按键或触摸事件在特定时间内无响应）

具体的超时时间的定义在 framework 下的

ActivityManagerService.java

//How long we wait until we timeout on key dispatching.

staticfinal int KEY_DISPATCHING_TIMEOUT = 5*1000

四：为什么会超时呢？

超时时间的计数一般是从按键分发给 app 开始。超时的原因一般有两种：

(1)当前的事件没有机会得到处理（即 UI 线程正在处理前一个事件，没有及时的完成或者 looper 被某种原因阻塞住了）

(2)当前的事件正在处理，但没有及时完成

五：如何避免 **KeyDispatchTimeout**

1：UI 线程尽量只做跟 UI 相关的工作

2：耗时的工作（比如数据库操作，I/O，连接网络或者别的有可能阻碍 UI 线程的操作）把它放入单独的线程处理

3：尽量用 Handler 来处理 UIthread 和别的 thread 之间的交互

六：UI 线程

说了那么多的 UI 线程，那么哪些属于 UI 线程呢？

UI 线程主要包括如下：

1. **Activity:onCreate(), onResume(), onDestroy(), onKeyDown(), onClick(),etc**
2. AsyncTask: onPreExecute(), onProgressUpdate(), onPostExecute(), onCancel,etc
3. Mainthread handler: handleMessage(), post*(runnable r), etc
4. other

七:如何去分析 ANR

先看个 LOG:

04-01 13:12:11.572 **I/InputDispatcher(220): Application is not responding**:Window{2b263310com.android.email/com.android.email.activity.SplitScreenActivitypaused=false}. 5009.8ms since event, 5009.5ms since waitstarted

04-0113:12:11.572 I/WindowManager(220): Input event dispatching timedout sending tocom.android.email/com.android.email.activity.SplitScreenActivity

04-01 **13:12:14.123 I/Process(220): Sending signal. PID: 21404 SIG: 3---发生 ANR 的时间和生成 trace.txt 的时间**

04-01 13:12:14.123 I/dalvikvm(21404):threadid=4: reacting to signal 3

……

04-0113:12:15.872 E/ActivityManager(220): ANR in com.android.email(com.android.email/.activity.SplitScreenActivity)

04-0113:12:15.872 E/ActivityManager(220): Reason:keyDispatchingTimedOut

04-0113:12:15.872 E/ActivityManager(220): Load: 8.68 / 8.37 / 8.53

04-0113:12:15.872 E/ActivityManager(220):**CPUusage from 4361ms to 699ms ago**----CPU 在 ANR 发生前的使用情况

04-0113:12:15.872 E/ActivityManager(220): 5.5%21404/com.android.email: 1.3% user + 4.1% kernel / faults: 10 minor

04-0113:12:15.872 E/ActivityManager(220): 4.3%220/system_server: 2.7% user + 1.5% kernel / faults: 11 minor 2 major

04-0113:12:15.872 E/ActivityManager(220): 0.9%52/spi_qsd.0: 0% user + 0.9% kernel

04-0113:12:15.872 E/ActivityManager(220): 0.5%65/irq/170-cyttsp-: 0% user + 0.5% kernel

04-0113:12:15.872 E/ActivityManager(220): 0.5%296/com.android.systemui: 0.5% user + 0% kernel

04-0113:12:15.872 E/ActivityManager(220): **100%TOTAL: 4.8% user + 7.6% kernel + 87% iowait**

04-0113:12:15.872 E/ActivityManager(220):**CPUusage from 3697ms to 4223ms later**:-- ANR 后 CPU 的使用量

04-0113:12:15.872 E/ActivityManager(220): 25%21404/com.android.email: 25% user + 0% kernel / faults: 191 minor

04-0113:12:15.872 E/ActivityManager(220): 16% 21603/__eas(par.hakan: 16% user + 0% kernel

04-0113:12:15.872 E/ActivityManager(220): 7.2% 21406/GC: 7.2% user + 0% kernel

04-0113:12:15.872 E/ActivityManager(220): 1.8% 21409/Compiler: 1.8% user + 0% kernel

04-0113:12:15.872 E/ActivityManager(220): 5.5%220/system_server: 0% user + 5.5% kernel / faults: 1 minor

04-0113:12:15.872 E/ActivityManager(220): 5.5% 263/InputDispatcher: 0% user + 5.5% kernel

04-0113:12:15.872 E/ActivityManager(220): **32%TOTAL: 28% user + 3.7% kernel**

从 LOG 可以看出 ANR 的类型，CPU 的使用情况，如果 CPU 使用量接近 100%，说明当前设备很忙，有可能是 CPU 饥饿导致了 ANR

如果 CPU 使用量很少，说明主线程被 BLOCK 了

如果 IOwait 很高，说明 ANR 有可能是主线程在进行 I/O 操作造成的

除了看 LOG，解决 ANR 还得需要 trace.txt 文件，

如何获取呢？可以用如下命令获取

1. $chmod 777 /data/anr
2. $rm /data/anr/traces.txt
3. $ps
4. $kill -3*PID*
5. *adbpull data/anr/traces.txt ./mytraces.txt*

从 trace.txt 文件，看到最多的是如下的信息：

-----pid 21404 at 2011-04-01**13:12:14** -----
Cmdline: com.android.email

DALVIK THREADS:
(mutexes: tll=0tsl=0 tscl=0 ghl=0 hwl=0 hwll=0)
"main" prio=5 tid=1**NATIVE**
  | group="main" sCount=1 dsCount=0obj=0x2aad2248 self=0xcf70
  | sysTid=21404 nice=0 sched=0/0cgrp=[fopen-error:2] handle=1876218976
  atandroid.os.MessageQueue.nativePollOnce(Native Method)
  atandroid.os.MessageQueue.next(MessageQueue.java:119)
  atandroid.os.Looper.loop(Looper.java:110)
 at android.app.ActivityThread.main(ActivityThread.java:3688)
 at java.lang.reflect.Method.invokeNative(Native Method)
  atjava.lang.reflect.Method.invoke(Method.java:507)
  atcom.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:866)
 at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:624)
 at dalvik.system.NativeStart.main(Native Method)

说明主线程在等待下条消息进入消息队列

八：Thread 状态

ThreadState (defined at “dalvik/vm/thread.h “)

THREAD_UNDEFINED = -1, /* makes enum compatible with int32_t */

THREAD_ZOMBIE = 0, /* TERMINATED */

THREAD_**RUNNING** = 1, /* RUNNABLE or running now */

THREAD_**TIMED_WAIT** = 2, /* TIMED_WAITING in Object.wait() */

THREAD_**MONITOR** = 3, /* BLOCKED on a monitor */

THREAD_**WAIT** = 4, /* WAITING in Object.wait() */

THREAD_INITIALIZING= 5, /* allocated, not yet running */

THREAD_STARTING = 6, /* started, not yet on thread list */

THREAD_**NATIVE** = 7, /* off in a JNI native method */

THREAD_**VMWAIT** = 8, /* waiting on a VM resource */

THREAD_**SUSPENDED** = 9, /* suspended, usually by GC or debugger */

九：如何调查并解决 ANR

1：首先分析 log

2: 从 trace.txt 文件查看调用 stack.

3: 看代码

4：仔细查看 ANR 的成因（iowait?block?memoryleak?）

十：案例

案例 1：关键词:ContentResolver in AsyncTask onPostExecute, high iowait

Process:com.android.email
Activity:com.android.email/.activity.MessageView
Subject:keyDispatchingTimedOut
CPU usage from 2550ms to -2814ms ago:
5%187/system_server: 3.5% user + 1.4% kernel / faults: 86 minor 20major
4.4% 1134/com.android.email: 0.7% user + 3.7% kernel /faults: 38 minor 19 major
4% 372/com.android.eventstream: 0.7%user + 3.3% kernel / faults: 6 minor
1.1% 272/com.android.phone:0.9% user + 0.1% kernel / faults: 33 minor
0.9%252/com.android.systemui: 0.9% user + 0% kernel
0%409/com.android.eventstream.telephonyplugin: 0% user + 0% kernel /faults: 2 minor
0.1% 632/com.android.devicemonitor: 0.1% user + 0%kernel
100%TOTAL: 6.9% user + 8.2% kernel +84%iowait

-----pid 1134 at 2010-12-17 17:46:51 -----
Cmd line:com.android.email

DALVIK THREADS:
(mutexes: tll=0 tsl=0tscl=0 ghl=0 hwl=0 hwll=0)
"main" prio=5 tid=1 WAIT
|group="main" sCount=1 dsCount=0 obj=0x2aaca180self=0xcf20
| sysTid=1134 nice=0 sched=0/0 cgrp=[fopen-error:2]handle=1876218976
at java.lang.Object.wait(Native Method)
-waiting on <0x2aaca218> (a java.lang.VMThread)
atjava.lang.Thread.parkFor(Thread.java:1424)
atjava.lang.LangAccessImpl.parkFor(LangAccessImpl.java:48)
atsun.misc.Unsafe.park(Unsafe.java:337)
atjava.util.concurrent.locks.LockSupport.park(LockSupport.java:157)
atjava.util.concurrent.locks.AbstractQueuedSynchronizer.parkAndCheckInterrupt(AbstractQueuedSynchronizer.java:808)
atjava.util.concurrent.locks.AbstractQueuedSynchronizer.acquireQueued(AbstractQueuedSynchronizer.java:841)
atjava.util.concurrent.locks.AbstractQueuedSynchronizer.acquire(AbstractQueuedSynchronizer.java:1171)
atjava.util.concurrent.locks.ReentrantLock$FairSync.lock(ReentrantLock.java:200)
atjava.util.concurrent.locks.ReentrantLock.lock(ReentrantLock.java:261)
atandroid.database.sqlite.SQLiteDatabase.lock(SQLiteDatabase.java:378)
atandroid.database.sqlite.SQLiteCursor.<init>(SQLiteCursor.java:222)
atandroid.database.sqlite.SQLiteDirectCursorDriver.query(SQLiteDirectCursorDriver.java:53)
atandroid.database.sqlite.SQLiteDatabase.rawQueryWithFactory(SQLiteDatabase.java:1356)
atandroid.database.sqlite.SQLiteDatabase.queryWithFactory(SQLiteDatabase.java:1235)
atandroid.database.sqlite.SQLiteDatabase.query(SQLiteDatabase.java:1189)
atandroid.database.sqlite.SQLiteDatabase.query(SQLiteDatabase.java:1271)
atcom.android.email.provider.EmailProvider.query(EmailProvider.java:1098)
atandroid.content.ContentProvider$Transport.query(ContentProvider.java:187)
atandroid.content.**ContentResolver.query**(ContentResolver.java:268)
atcom.android.email.provider.EmailContent$Message.restoreMessageWithId(EmailContent.java:648)
atcom.android.email.Controller.setMessageRead(Controller.java:658)
atcom.android.email.activity.MessageView.onMarkAsRead(MessageView.java:700)
atcom.android.email.activity.MessageView.access$2500(MessageView.java:98)
at**com.android.email.activity.MessageView$LoadBodyTask.onPostExecute**(MessageView.java:1290)
atcom.android.email.activity.MessageView$LoadBodyTask.onPostExecute(MessageView

.java:1255)
atandroid.os.AsyncTask.finish(AsyncTask.java:417)
atandroid.os.AsyncTask.access$300(AsyncTask.java:127)
at**android.os.AsyncTask$InternalHandler.handleMessage**(AsyncTask.java:429)
atandroid.os.Handler.dispatchMessage(Handler.java:99)
atandroid.os.Looper.loop(Looper.java:123)
atandroid.app.ActivityThread.main(ActivityThread.java:3652)
atjava.lang.reflect.Method.invokeNative(Native Method)
atjava.lang.reflect.Method.invoke(Method.java:507)
atcom.android.internal.os.ZygoteIn

原因：IOWait 很高，说明当前系统在忙于 I/O，因此数据库操作被阻塞

原来：

```
finalMessagemessage=Message.restoreMessageWithId(mProviderContext,messageId);
if(message==null){
   return;
}

Accountaccount=Account.restoreAccountWithId(mProviderContext,message.mAccountKe
y);

if(account==null){
   return;//isMessagingController returns false for null, but let's make itclear.
}

if(isMessagingController(account)){
   new Thread(){
       @Override
       public void run(){
          mLegacyController.processPendingActions(message.mAccountKey);
       }
   }.start();
}
```

解决后：

```
newThread() {
    finalMessagemessage=Message.restoreMessageWithId(mProviderContext,messageId
);

    if(message==null){
        return;
    }

    Accountaccount=Account.restoreAccountWithId(mProviderContext,message.mAccou
```

```
ntKey);

    if(account==null){
       return;//isMessagingController returns false for null, but let's make
itclear.
    }

    if(isMessagingController(account)) {
        mLegacyController.processPendingActions(message.mAccountKey);
    }
}.start();
```

关于 AsyncTask:http://developer.android.com/reference/android/os/AsyncTask.html

案例 2：关键词:在 UI 线程进行网络数据的读写

ANRin process: com.android.mediascape:PhotoViewer (last
incom.android.mediascape:PhotoViewer)
Annotation:keyDispatchingTimedOut
CPU usage:
Load: 6.74 / 6.89 / 6.12
CPUusage from 8254ms to 3224ms ago:
ovider.webmedia: 4% = 4% user +0% kernel / faults: 68 minor
system_server: 2% = 1% user + 0%kernel / faults: 18 minor
re-initialized>: 0% = 0% user + 0%kernel / faults: 50 minor
events/0: 0% = 0% user + 0%kernel
TOTAL:7% = 6% user + 1% kernel

DALVIKTHREADS:
""main"" prio=5 tid=3 NATIVE
|group=""main"" sCount=1 dsCount=0 s=Yobj=0x4001b240 self=0xbda8
| sysTid=2579 nice=0 sched=0/0cgrp=unknown handle=-1343993184
atorg.apache.harmony.luni.platform.OSNetworkSystem.receiveStreamImpl(NativeMeth
od)
atorg.apache.harmony.luni.platform.**OSNetworkSystem.receiveStream**(OSNetworkSyst
em.java:478)
atorg.apache.harmony.luni.net.PlainSocketImpl.read(PlainSocketImpl.java:565)
atorg.apache.harmony.luni.net.SocketInputStream.read(SocketInputStream.java:87)
atorg.apache.harmony.luni.internal.net.www.protocol.http.HttpURLConnection$Limite
dInputStream.read(HttpURLConnection.java:303)
atjava.io.InputStream.read(InputStream.java:133)
atjava.io.BufferedInputStream.fillbuf(BufferedInputStream.java:157)
atjava.io.BufferedInputStream.read(BufferedInputStream.java:346)

atandroid.graphics.BitmapFactory.nativeDecodeStream(Native Method)
atandroid.graphics.**BitmapFactory.decodeStream**(BitmapFactory.java:459)
atcom.android.mediascape.activity.PhotoViewerActivity.**getPreviewImage**(PhotoViewerActivity.java:4465)
atcom.android.mediascape.activity.PhotoViewerActivity.**dispPreview**(PhotoViewerActivity.java:4406)
atcom.android.mediascape.activity.PhotoViewerActivity.access$6500(PhotoViewerActivity.java:125)
at**com.android.mediascape.activity.PhotoViewerActivity$33$1.run**(PhotoViewerActivity.java:4558)
atandroid.os.Handler.handleCallback(Handler.java:587)
atandroid.os.Handler.dispatchMessage(Handler.java:92)
atandroid.os.Looper.loop(Looper.java:123)
atandroid.app.ActivityThread.main(ActivityThread.java:4370)
atjava.lang.reflect.Method.invokeNative(Native Method)
atjava.lang.reflect.Method.invoke(Method.java:521)
atcom.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:868)
atcom.android.internal.os.ZygoteInit.main(ZygoteInit.java:626)
atdalvik.system.NativeStart.main(Native Method)

关于网络连接，在设计的时候可以设置个 timeout 的时间或者放入独立的线程来处理。

关于 Handler 的问题，可以参考：
http://developer.android.com/reference/android/os/Handler.html

案例 3：

关键词：Memoryleak/Thread leak

11-1621:41:42.560 I/ActivityManager(1190): ANR in process:android.process.acore (last in android.process.acore)
11-1621:41:42.560 I/ActivityManager(1190): Annotation:keyDispatchingTimedOut
11-16 21:41:42.560 I/ActivityManager(1190): CPU usage:
11-16 21:41:42.560 I/ActivityManager(1190):Load: 11.5 / 11.1 / 11.09
11-16 21:41:42.560 I/ActivityManager(1190): CPU usage from 9046ms to 4018ms ago:
11-16 21:41:42.560I/ActivityManager(1190): **d.process.acore:98%**= 97% user + 0% kernel / faults: 1134 minor
11-16 21:41:42.560I/ActivityManager(1190): system_server: 0% = 0% user + 0% kernel /faults: 1 minor
11-16 21:41:42.560 I/ActivityManager(1190): adbd:0% = 0% user + 0% kernel
11-16 21:41:42.560 I/ActivityManager(1190): logcat: 0% = 0% user + 0% kernel
11-16 21:41:42.560I/ActivityManager(1190): **TOTAL:100% = 98% user + 1% kernel**

Cmdline: android.process.acore

DALVIK THREADS:
"main"prio=5 tid=3 **VMWAIT**
|group="main" sCount=1 dsCount=0 s=N obj=0x40026240self=0xbda8
| sysTid=1815 nice=0 sched=0/0 cgrp=unknownhandle=-1344001376
atdalvik.system.**VMRuntime.trackExternalAllocation**(NativeMethod)
atandroid.graphics.Bitmap.nativeCreate(Native Method)
atandroid.graphics.**Bitmap.createBitmap**(Bitmap.java:468)
atandroid.view.View.buildDrawingCache(View.java:6324)
atandroid.view.View.getDrawingCache(View.java:6178)
atandroid.view.ViewGroup.drawChild(ViewGroup.java:1541)
……
atcom.android.internal.policy.impl.PhoneWindow$DecorView.draw(PhoneWindow.java:1830)
atandroid.view.ViewRoot.draw(ViewRoot.java:1349)
atandroid.view.ViewRoot.performTraversals(ViewRoot.java:1114)
atandroid.view.ViewRoot.handleMessage(ViewRoot.java:1633)
atandroid.os.Handler.dispatchMessage(Handler.java:99)
atandroid.os.Looper.loop(Looper.java:123)
atandroid.app.ActivityThread.main(ActivityThread.java:4370)
atjava.lang.reflect.Method.invokeNative(Native Method)
atjava.lang.reflect.Method.invoke(Method.java:521)
atcom.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:868)
atcom.android.internal.os.ZygoteInit.main(ZygoteInit.java:626)
atdalvik.system.NativeStart.main(Native Method)

"Thread-408"prio=5 tid=329 WAIT
|group="main" sCount=1 dsCount=0 s=N obj=0x46910d40self=0xcd0548
| sysTid=10602 nice=0 sched=0/0 cgrp=unknownhandle=15470792
at java.lang.Object.wait(Native Method)
-waiting on <0x468cd420> (a java.lang.Object)
atjava.lang.Object.wait(Object.java:288)
atcom.android.dialer.CallLogContentHelper$UiUpdaterExecutor$1.run(CallLogContentHelper.java:289)
atjava.lang.Thread.run(Thread.java:1096)

分析：

atdalvik.system.**VMRuntime.trackExternalAllocation**(NativeMethod)内存不足导致 block 在创建 bitmap 上

**MEMINFO in pid 1360 [android.process.acore] **
native dalvik other total
size: 17036 **23111** N/A 40147
allocated: 16484 20675 N/A 37159
free: 296 2436 N/A 2732

解决：如果机器的内存族，可以修改虚拟机的内存为 36M 或更大，不过最好是复查代码，查看哪些内存没有释放