When using Android Studio debugger on a background thread, my Android app crashes when stepping through a method that contains references to DisplayMetrics, even thought the DisplayMetrics code is never hit

+1 2   Hotlists (1)   Mark as Duplicate

Comments (10)   Dependencies   Duplicates (0)   Blocking (0)   Resources (4)

Infeasible   Bug   P1   + Add Hotlist

STATUS UPDATE No update yet. Edit

Reporter	mm...@twobitlabs.com
Type	Bug
Priority	P1
Severity	S3
Status	Won't fix (Infeasible)
Access	Default access View
Assignee	sh...@google.com
Verifier	--
Collaborators	
CC	mm...@twobitlabs.com vs...@google.com
AOSP ID	68642
ReportedBy	Developer
Found In	--
Targeted To	--
Verified In	--
In Prod	

DESCRIPTION mm...@twobitlabs.com created issue #1   Apr 17, 2014 06:02AM

Version of Android Studio (available in the about box): 0.5.3, Build #AI-135.1092050

OS version: Windows 8.1 Pro
Java JRE/JDK version: 1.7.0_51-b13 amd64

I have a method, isTablet():

```
  public static boolean isTablet() {
     if (!isInitialized) {
        try {
           // Compute screen size
           DisplayMetrics dm = TsApplication.getAppContext().getResources().getDisplayMetrics();
           float screenWidth  = dm.widthPixels / dm.xdpi;
           float screenHeight = dm.heightPixels / dm.ydpi;
           double diagonal = Math.sqrt(Math.pow(screenWidth, 2) + Math.pow(screenHeight, 2));
           // Tablet devices should have a screen size of at least 10 inches
           isTablet = diagonal >= 10.0;
        } catch(Throwable t) {
           Log.e(LOG_TAG, "Failed to compute screen size", t);
           isTablet = false;
        } finally {
           isInitialized = true;
        }
     }
     return isTablet;
  }
```

Assume that isInitialized is true, and isTablet() was invoked on a background thread. I have a breakpoint on the first line of the method. If I hit that breakpoint, then hit F9 (resume program), the value of isTablet is returned and the program proceeds merrily along. However, if I instead hit F8 (step over), to advance to the return statement at the bottom of the method (since isInitialied is true), the app that is being debugged immediately crashes. The following line is written to logcat:

```
04-16 15:00:35.926    7637-7645/com.identifyingdetailsremoved.foo A/libc? Fatal signal 11 (SIGSEGV) at
0x000d2110 (code=1), thread 7645 (JDWP)
```

If I move the entire try block into a separate method, and update isTablet() to call that method, like this...

```
  public static boolean isTablet() {
     if (isTablet == null) {
        initializeDeviceType();
     }
     return isTablet;
  }
```

... then there's no crash. I can step from the first line of the method to that last, and then keep stepping into the method that called isTablet(), even if it was invoked on a background thread.

I suspect, in the case of the crash, that the debugger is doing more than it needs to, and is attempting to load information related to DisplayMetrics, even though the conditional logic completely sidesteps the code related to DisplayMetrics. And there must be something about accessing DisplayMetrics from a background thread that is problematic.

✓ Links (4)   Hide all

"https://github.com/nmr8acme/dalvik-debug-bug"   nm...@ #4

"Log from phone while crashing: https://gist.github.com/nmr8acme/7d1b482bbd886a294b3d"	nm...@ #4
" - https://code.google.com/p/android-developer-preview/issues/… "	nm...@ #9
" - https://code.google.com/p/android-developer-preview/issues/… "	nm...@ #9

COMMENTS

All comments ▾ ↓ Oldest first

vs...@google.com <vs...@google.com> #2 Apr 17, 2014 08:01AM

Assigned to vs...@google.com.

1. Is it possible to provide a reproducible test case? I'll try copying the code as is and see if I can reproduce it, but it would make it a lot easier if the problem can be reproduced.

2. Could you provide some details on the device? (emulator or device? android version?)

It seems unlikely that this is because of a background thread, but more of an issue of IntelliJ requesting some data which seems to crash the JVM (I assume Dalvik). Is there a longer stack trace or is it just that one line? The JDWP at the end seems to suggest that it is the VM thread that talks to the debugger that crashed.

mm...@twobitlabs.com <mm...@twobitlabs.com> #3 Apr 17, 2014 08:09AM

1. I'll work on providing a complete, reproducible test case when I have some time, and will post that in a comment.

2. This happened on GenyMotion (emulating a Samsung Galaxy S3) and an actual device (Samsung Galaxy S3), both running API 18.

> It seems unlikely that this is because of a background thread...

Just to clarify, because I'm not sure what you meant by the above comment... the problem only occurs when the isTablet() method is invoked on a background thread. When isTablet() is invoked on the main thread, I can step through just fine.

I haven't been able to come up with a stack trace -- just that one-line fatal log message.

Thanks for the quick response!

nm...@gmail.com <nm...@gmail.com> #4 Aug 20, 2014 09:20AM

Here's a tiny repro for a very similar bug

https://github.com/nmr8acme/dalvik-debug-bug

Maybe it has something to do with multiple return points from the function? Or DM has badly behaved JNI code?

Log from phone while crashing: https://gist.github.com/nmr8acme/7d1b482bbd886a294b3d

Nexus 5 build KTU84P (4.4.4)

Android Studio 0.8.6

vs...@google.com <vs...@google.com> #5 Aug 20, 2014 09:22AM

Thanks for the test case, I'll try it out this week.

nm...@gmail.com <nm...@gmail.com> #6 Aug 20, 2014 09:23AM

Not sure what Android build tools Gradle would build this with, but according to the SDK manager I have a bunch installed.

deleted Restricted
0 B ?

nm...@gmail.com <nm...@gmail.com> #7 Aug 20, 2014 09:30AM

[Comment deleted]

deleted Restricted

0 B

vs...@google.com <vs...@google.com> #8 Aug 29, 2014 04:39AM

Reassigned to sh...@google.com.

I can reproduce the crash, which is in Dalvik.

```
I/DEBUG   (  249): backtrace:
I/DEBUG   (  249):     #00  pc 00022114  /system/lib/libc.so (tgkill+12)
I/DEBUG   (  249):     #01  pc 00013169  /system/lib/libc.so (pthread_kill+48)
I/DEBUG   (  249):     #02  pc 0001337d  /system/lib/libc.so (raise+10)
I/DEBUG   (  249):     #03  pc 000120b3  /system/lib/libc.so
I/DEBUG   (  249):     #04  pc 000219c8  /system/lib/libc.so (abort+4)
I/DEBUG   (  249):     #05  pc 000493df  /system/lib/libdvm.so (dvmAbort+78)
I/DEBUG   (  249):     #06  pc 00046b23  /system/lib/libdvm.so (dvmDbgGetLocalValue(unsigned long
long, unsigned long long, int, unsigned char, unsigned char*, int)+278)
I/DEBUG   (  249):     #07  pc 00066def  /system/lib/libdvm.so
I/DEBUG   (  249):     #08  pc 0006753b  /system/lib/libdvm.so (dvmJdwpProcessRequest(JdwpState*,
JdwpReqHeader const*, unsigned char const*, int, ExpandBuf*)+110)
I/DEBUG   (  249):     #09  pc 00064e1d  /system/lib/libdvm.so
I/DEBUG   (  249):     #10  pc 0006775d  /system/lib/libdvm.so
I/DEBUG   (  249):     #11  pc 000580d5  /system/lib/libdvm.so
I/DEBUG   (  249):     #12  pc 0000d258  /system/lib/libc.so (__thread_entry+72)
I/DEBUG   (  249):     #13  pc 0000d3f0  /system/lib/libc.so (pthread_create+240)
```

This seems to be fixed in L with the ART runtime. It is unlikely that Dalvik itself will be fixed.

nm...@gmail.com <nm...@gmail.com> #9 Aug 29, 2014 07:18AM

Argh

AFAICT ART is full of it's own set of debugger crashes
 - https://code.google.com/p/android-developer-preview/issues/detail?id=1135
 - https://code.google.com/p/android-developer-preview/issues/detail?id=1136

This makes Android debugging feel like a nice game of Russian roulette.

sh...@google.com <sh...@google.com> #10 Jun 26, 2015 07:45PM

Status: Won't Fix (Infeasible)

We no longer make any change to Dalvik. ART is the default runtime starting from Android 5.0 and does not seem impacted according to comment #7.

If you can reproduce with ART (on a device with Android 5.0+), please file another bug.