DESCRIPTION [Deleted User] created issue #1 Oct

This is on a nexus 5x running Nougat. It does not happen on any other phone in our lab. Also tried loading older versions of our app (from several months ago) and those versions are also now c
know it's not something that we added to the app recently.

Somebody on Stack Overflow seems to have the same problem.
http://stackoverflow.com/questions/40096361/native-crash-of-renderthread-with-signal-6-sigabrt?newreg=8af8e2270c9b4fab8bce30f7f43149ff

****
First we get this:
****

```
10-18 10:42:15.415 32304-32433/com.gogobot.gogodroid.mdev E/OpenGLRenderer: Texture is using 0.00B, count = 1
                                                                            OffscreenBuffer is using 0.00B, count = 0
                                                                            Layer is using 0.00B, count = 1
10-18 10:42:15.416 32304-32433/com.gogobot.gogodroid.mdev A/OpenGLRenderer: Leaked 2 GPU objects!
10-18 10:42:15.424 866-18913/? I/ActivityManager: Start proc 704:com.google.process.gapps/u0a13 for content provider com.google.android.gsf/.settings.GoogleSettingsProvider
10-18 10:42:15.425 32304-32433/com.gogobot.gogodroid.mdev A/libc: Fatal signal 6 (SIGABRT), code -6 in tid 32433 (RenderThread)

                                                                  [ 10-18 10:42:15.429   364:  364 W/       ]
                                                                  debuggerd: handling request: pid=32304 uid=10120 gid=10120 tid=32433
```

****
Then we get this:
****

```
10-18 10:42:15.806 715-715/? A/DEBUG: *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
10-18 10:42:15.806 715-715/? A/DEBUG: Build fingerprint: 'google/bullhead/bullhead:7.0/NBD90W/3239497:user/release-keys'
10-18 10:42:15.806 715-715/? A/DEBUG: Revision: 'rev_1.0'
10-18 10:42:15.806 715-715/? A/DEBUG: ABI: 'arm64'
10-18 10:42:15.806 715-715/? A/DEBUG: pid: 32304, tid: 32433, name: RenderThread  >>> com.gogobot.gogodroid.mdev <<<
10-18 10:42:15.806 715-715/? A/DEBUG: signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
10-18 10:42:15.825 715-715/? A/DEBUG: Abort message: 'SANDBOX: Adjust is running in Sandbox mode. Use this setting for testing. Don't forget to set the environment to `production` before pu
10-18 10:42:15.825 715-715/? A/DEBUG:    x0  0000000000000000 x1  0000000000007eb1 x2  0000000000000006 x3  0000000000000008
10-18 10:42:15.825 715-715/? A/DEBUG:    x4  00006e6174736e69 x5  0000000000000000 x6  0000007610dbb000 x7  0000000000000000
10-18 10:42:15.825 715-715/? A/DEBUG:    x8  0000000000000083 x9  ffffffffffffffdf x10 0000000000000000 x11 0000000000000001
10-18 10:42:15.825 715-715/? A/DEBUG:    x12 0000000000000018 x13 0000000000000000 x14 0000000000000000 x15 0018ce134ebca82c
10-18 10:42:15.825 715-715/? A/DEBUG:    x16 000000760e80fed0 x17 000000760e7b9a2c x18 0000000000000400 x19 00000075ebce64f8
10-18 10:42:15.826 715-715/? A/DEBUG:    x20 0000000000000006 x21 00000075ebce6450 x22 000000000000000b x23 0000007610197040
10-18 10:42:15.826 715-715/? A/DEBUG:    x24 00000000ffffffff x25 00000075d0393280 x26 7fffffffffffffff x27 00000075ee2896b0
10-18 10:42:15.826 715-715/? A/DEBUG:    x28 000000760d148878 x29 00000075ebce5ae0 x30 000000760e7b6e58
10-18 10:42:15.826 715-715/? A/DEBUG:    sp  00000075ebce5ac0 pc  000000760e7b9a34 pstate 0000000060000000
10-18 10:42:15.849 715-715/? A/DEBUG: backtrace:
10-18 10:42:15.849 715-715/? A/DEBUG:    #00 pc 0000000000006ba34  /system/lib64/libc.so (tgkill+8)
10-18 10:42:15.849 715-715/? A/DEBUG:    #01 pc 0000000000068e54  /system/lib64/libc.so (pthread_kill+64)
10-18 10:42:15.849 715-715/? A/DEBUG:    #02 pc 0000000000023ed8  /system/lib64/libc.so (raise+24)
10-18 10:42:15.849 715-715/? A/DEBUG:    #03 pc 000000000001c790  /system/lib64/libc.so (abort+52)
10-18 10:42:15.849 715-715/? A/DEBUG:    #04 pc 00000000000107f4  /system/lib64/libcutils.so (__android_log_assert+224)
10-18 10:42:15.850 715-715/? A/DEBUG:    #05 pc 0000000000054cc0  /system/lib64/libhwui.so
10-18 10:42:15.850 715-715/? A/DEBUG:    #06 pc 00000000000350f8  /system/lib64/libhwui.so
10-18 10:42:15.850 715-715/? A/DEBUG:    #07 pc 00000000000328c8  /system/lib64/libhwui.so
10-18 10:42:15.850 715-715/? A/DEBUG:    #08 pc 0000000000037348  /system/lib64/libhwui.so
10-18 10:42:15.850 715-715/? A/DEBUG:    #09 pc 0000000000038434  /system/lib64/libhwui.so
10-18 10:42:15.850 715-715/? A/DEBUG:    #10 pc 0000000000039890  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+152)
10-18 10:42:15.850 715-715/? A/DEBUG:    #11 pc 0000000000012460  /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
10-18 10:42:15.850 715-715/? A/DEBUG:    #12 pc 000000000009bc4c  /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
10-18 10:42:15.850 715-715/? A/DEBUG:    #13 pc 0000000000006863c  /system/lib64/libc.so (_ZL15__pthread_startPv+208)
10-18 10:42:15.850 715-715/? A/DEBUG:    #14 pc 000000000001d9fc  /system/lib64/libc.so (__start_thread+16)
```

✓ Mentioned issues (3) ✓ Links (79)

Mentioned issues (3)

-- -- “https://issuetracker.google.com/32322297”

-- -- “https://issuetracker.google.com/62984068”

P4 Occasionally happening tgkill on Android 7 "https://issuetracker.google.com/71938053"

Links (79)

"http://stackoverflow.com/questions/40096361/native-crash-of-renderthread-with-signal-6-sigabrt?newreg=8af8e2… "
"https://drive.google.com/file/d/0Bzu4wPnZXr0FaTFFejM2STFkVTA/… "
"https://play.google.com/store/apps/details?id=com.gog… "
"http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/Gpu… "
" …Tracker::onGLContextDestroyed() is static and it is called from RenderState::onGLContextDestroyed() http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderstate/RenderState.cp

See all related links

COMMENTS

All comments

nm...@gmail.com <nm...@gmail.com> #2 Oct 19

I'm also seeing this issue, specifically under low-memory conditions and only on a Nexus 5x running Nougat. I was trying to launch a camera intent from a fragment. I've attached my log as w

deleted
0 B

[Deleted User] <[Deleted User]> #3 Oct 19

I'm seeing this crash too, specifically on build NBD90W of Android on the nexus 5x (october security patches). We're not using a camera intent, but we're using the camera API directly.

nm...@gmail.com <nm...@gmail.com> #4 Oct 19

Oh, good point -- I, too, am running build NBD90W with the October 5 security patches.

ga...@gmail.com <ga...@gmail.com> #5 Oct 19

Same here and my app uses Google Maps and a few images. It only crashes upon exit. My device is also Nexus 5X w/ build number NBD90W.

[Deleted User] <[Deleted User]> #6 Oct 19

I am on NBD90W as well.

sl...@gmail.com <sl...@gmail.com> #7 Oct 19

I was the reporter for the question on Stackoverflow -- I'm definitely glad other people are having this issue. Hopefully it can be resolved!

ar...@google.com <ar...@google.com> #8 Oct 19

Assigned to ar...@google.com.

Can you provide the below requested information to better understand the issue:

Steps to reproduce
What steps do others need to take in order to reproduce the issue themselves?
Please attach sample project to reproduce the issue.

Frequency
How frequently does this issue occur? (e.g 100% of the time, 10% of the time)

Expected output
What do you expect to occur?

Current output
What do you see instead?

Android bug report:
After reproducing the issue, navigate to developer settings, ensure 'USB debugging' is enabled, then enable 'Bug report shortcut'. To take bug report, hold the power button and select the 'Tak

Note: Please upload the files to google drive and share the folder to android-bugreport@google.com, then share the link here.

[Deleted User] <[Deleted User]> #9 Oct 19

Seeing the same crash here, I am using Android 7 and Nexus 5x.

nm...@gmail.com <nm...@gmail.com> #10 Oct 20

I can't come up with a good example project and unfortunately, I cannot share my project publicly as a result of my company's NDA. But, here's the bug report from our app:

https://drive.google.com/file/d/0Bzu4wPnZXr0FaTFFejM2STFkVTA/view?usp=sharing

I shared the file with android-bugreport@google.com as well but it looks like the email failed to deliver.

In any case, I have a fragment in my application. When an activity uses the fragment (via supportFragmentManager), things work fine. But when another activity has the fragment contained ir crash around 80% of the time. I experience the crash whenever I background this fragment. It should be noted that the app uses very similar amounts of memory in both of these cases (~30M

The issue certainly seems to be memory-related. When I background the fragment, I get the standard TRIM_MEMORY_UI_HIDDEN from onTrimMemory. But just a few seconds later, I get a TRIM_MEMORY_COMPLETE and the app is killed (see my previous log).

sp...@gmail.com <sp...@gmail.com> #11 Oct 2(

[Comment deleted]

[Deleted User] <[Deleted User]> #12 Oct 2(

You can download our app from the Play Store (Gogobot City and Travel Guide, link below) and put it on your Nexus 5x with the latest updates. Run our app. Tap into some of the restaurants and generally exercise the app (but please don't put up any fake ratings, just browse - this is our live production app). Then exit out of the app by pressing the back button. You will get a mes "Gogobot has stopped - x Close app" after a few seconds.

https://play.google.com/store/apps/details?id=com.gogobot.gogodroid

[Deleted User] <[Deleted User]> #13 Oct 2(

It sounds like the issue is in the Nexus 5X's NBD90W Security update (firmware)- given that apps only started having this issue after the update was pushed.

Is this the right place to report a bug in that firmware or does Google use a different tracker for firmware (of so can you move it). The team that wrote that firmware update needs to look at t (probably with some urgency) as they seem widespread.

th...@gmail.com <th...@gmail.com> #14 Oct 2(

I'm having the same issue on app that I'm developing. In my view which extends TextureView and I have mediaPlayer inside which plays video. I lost couple of days trying to figure out what is onPause in fragment I've nulled the view, and I still get this error. :-/ Nexus 5x, Nouget.

ar...@google.com <ar...@google.com> #15 Oct 2

We have passed this defect on to the development team and will update this issue with more information as it becomes available.

ga...@gmail.com <ga...@gmail.com> #16 Oct 2

I'm experiencing the same issue on a Nexus 5. It occurs randomly when my app has been paused for a while, and then a fragment with a camera is resumed and a viewpager with video. App a prompt that it's not responding.

I'm on the M4B30X build with the 5 October security patch.

Crash report:
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
Build fingerprint: 'google/angler/angler:7.0/NBD90X/3254009:user/release-keys'
Revision: '0'
ABI: 'arm64'
pid: 16569, tid: 17391, name: RenderThread >>> XXX <<<
signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
Abort message: 'Leaked 6 GPU objects!'
x0 0000000000000000 x1 00000000000043ef x2 0000000000000006 x3 0000000000000008
x4 00006e6174736e69 x5 0000000000000000 x6 000000735a5cd000 x7 0000000000000000
x8 0000000000000083 x9 ffffffffffffffdf x10 0000000000000000 x11 0000000000000001
x12 0000000000000018 x13 0000000000000000 x14 0000000000000000 x15 0002e1100a26d22f
x16 0000007359d9eed0 x17 0000007359d48a2c x18 0000000000000400 x19 000000733c9054f8
x20 0000000000000006 x21 000000733c905450 x22 0000000000000000 x23 0000007357a4d040
x24 00000000ffffffff x25 000000733c7b3a10 x26 7fffffffffffffff x27 00000073578bad04
x28 000000734ee842f8 x29 000000733c904ae0 x30 0000007359d45e58
sp 000000733c904ac0 pc 0000007359d48a34 pstate 0000000060000000

backtrace:
#00 pc 000000000006ba34 /system/lib64/libc.so (tgkill+8)
#01 pc 0000000000068e54 /system/lib64/libc.so (pthread_kill+64)
#02 pc 0000000000023ed8 /system/lib64/libc.so (raise+24)
#03 pc 000000000001c790 /system/lib64/libc.so (abort+52)
#04 pc 00000000000107f4 /system/lib64/libcutils.so (__android_log_assert+224)
#05 pc 0000000000054cc0 /system/lib64/libhwui.so
#06 pc 00000000000350f8 /system/lib64/libhwui.so
#07 pc 00000000000328c8 /system/lib64/libhwui.so
#08 pc 0000000000037348 /system/lib64/libhwui.so
#09 pc 0000000000038434 /system/lib64/libhwui.so
#10 pc 0000000000039890 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+152)
#11 pc 0000000000012460 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
#12 pc 000000000009bc4c /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
#13 pc 000000000006863c /system/lib64/libc.so (_ZL15__pthread_startPv+208)
#14 pc 000000000001d9fc /system/lib64/libc.so (__start_thread+16)

ga...@gmail.com <ga...@gmail.com> #17 Oct 2

[Comment deleted]

ga...@gmail.com <ga...@gmail.com> #18 Oct 2

Sorry that crash report above is from a Nexus 6P, I can't verify at the moment if the Nexus 5 also has this issue.

or...@gmail.com <or...@gmail.com> #19 Oct 2

Was seeing this on 5x as well, upgraded to developer preview of 7.1 and it doesn't seem to be happening anymore

[Deleted User] <[Deleted User]> #20 Oct 2

Not sure why this was prioritized as "small". Our production app, which has been out for quite a while, crashes when the user's phone is updated to NBD90W. This is NOT a small priority for

ga...@gmail.com <ga...@gmail.com> #21 Oct 3

Happens on a Nexus 6P with build NBD90X every single time user leaves app when camera was opened in a fragment.

al...@gmail.com <al...@gmail.com> #22 Oct 3

Happens on a Nexus 5x with the same build in random sections of my app, it has no sense.

be...@gmail.com <be...@gmail.com> #23 Nov

I am also seeing this on my Nexus 6P running 7.0. We are using Google Maps and also the camera. The app always crash when leaving the activity that has the map and the camera intent.

[Deleted User] <[Deleted User]> #24 Nov

Unfortunately seeing it occasionally on Pixel running 7.1:

*** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
Build fingerprint: 'google/sailfish/sailfish:7.1/NDE63P/3332229:user/release-keys'
Revision: '0'
ABI: 'arm64'
pid: 18129, tid: 18551, name: RenderThread >>> com.yesitsyes.android.wyd.prod <<<
signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
Abort message: 'Leaked 2 GPU objects!'

su...@gmail.com <su...@gmail.com> #25 Nov 4

Seeing the issue with Nexus 5x build NBD90W. It happens when i run intent to upload files and start camera to take a picture. It is consistent every time.

[Deleted User] <[Deleted User]> #26 Nov 5

I am seeing this crash as well. Nexus 5X running 7.0

*** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
Build fingerprint: 'google/bullhead/bullhead:7.0/NBD90W/3239497:user/release-keys'
Revision: 'rev_1.0'
ABI: 'arm64'
pid: 7774, tid: 7823, name: RenderThread >>> com.electricobjects.client <<<
signal 6 (SIGABRT), code 0 (SI_USER), fault addr --------
Abort message: 'Leaked 12 GPU objects!'
x0 0000000000000000 x1 0000000000001e8f x2 0000000000000006 x3 0000000000000008
x4 00006e6174736e69 x5 0000000000000000 x6 00000079488ad000 x7 0000000000000000
x8 0000000000000083 x9 ffffffffffffffdf x10 0000000000000000 x11 0000000000000001
x12 0000000000000018 x13 0000000000000000 x14 0000000000000000 x15 00325b4238c3b7a1
x16 00000079466c8ed0 x17 0000007946672a2c x18 0000000000000400 x19 00000079298fa4f8
x20 0000000000000006 x21 00000079298fa450 x22 000000000000000b x23 0000007947cb9040
x24 00000000ffffffff x25 0000007925a176a0 x26 7fffffffffffffff x27 000000792623ebc0
x28 0000007944b463b8 x29 00000079298f9ae0 x30 000000794666fe58
sp 00000079298f9ac0 pc 0000007946672a34 pstate 0000000060000000

backtrace:
#00 pc 000000000006ba34 /system/lib64/libc.so (tgkill+8)
#01 pc 0000000000068e54 /system/lib64/libc.so (pthread_kill+64)
#02 pc 0000000000023ed8 /system/lib64/libc.so (raise+24)
#03 pc 000000000001c790 /system/lib64/libc.so (abort+52)
#04 pc 00000000000107f4 /system/lib64/libcutils.so (__android_log_assert+224)
#05 pc 0000000000054cc0 /system/lib64/libhwui.so
#06 pc 00000000000350f8 /system/lib64/libhwui.so
#07 pc 00000000000328c8 /system/lib64/libhwui.so

#08 pc 0000000000037348 /system/lib64/libhwui.so
#09 pc 0000000000038434 /system/lib64/libhwui.so
#10 pc 0000000000039890 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+152)
#11 pc 0000000000012460 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
#12 pc 000000000009bc4c /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
#13 pc 000000000006863c /system/lib64/libc.so (_ZL15__pthread_startPv+208)
#14 pc 000000000001d9fc /system/lib64/libc.so (__start_thread+16)

[Deleted User] <[Deleted User]> #27 Nov 8

We've been able to get better symbols from that stack trace, which might help:

Module "libc.so", in tgkill + 0x8
Module "libc.so", in pthread_kill + 0x40
Module "libc.so", in raise + 0x18
Module "libc.so", in abort + 0x34
Module "libcutils.so", in __android_log_assert + 0xe0
Module "libhwui.so", in android::uirenderer::DisplayListCanvas::flushAndAddOp(android::uirenderer::DisplayListOp*) + 0x8080
Module "libhwui.so", in android::uirenderer::renderthread::CanvasContext::getFrameNumber() + 0x21c8
Module "libhwui.so", in android::uirenderer::Stencil::dump() + 0x23c0
Module "libhwui.so", in android::uirenderer::renderthread::RenderProxy::trimMemory(int) + 0x88
Module "libhwui.so", in android::uirenderer::renderthread::RenderProxy::copySurfaceInto(android::sp<android::Surface>&, SkBitmap*) + 0xe4
Module "libhwui.so", in android::uirenderer::renderthread::RenderThread::threadLoop() + 0x98
Module "libutils.so", in android::Thread::_threadLoop(void*) + 0x110
Module "libandroid_runtime.so", in android::AndroidRuntime::javaThreadShell(void*) + 0x74
Module "libc.so", in __pthread_start(void*) + 0xd0
Module "libc.so", in __start_thread + 0x10

ja...@gmail.com <ja...@gmail.com> #28 Nov 8

Symbolicated stack trace (with symbols for libhwui.so):

Module "libc.so", in tgkill + 0x8
Module "libc.so", in pthread_kill + 0x40
Module "libc.so", in raise + 0x18
Module "libc.so", in abort + 0x34
Module "libcutils.so", in __android_log_assert + 0xe0
Module "libhwui.so", in android::uirenderer::DisplayListCanvas::flushAndAddOp(android::uirenderer::DisplayListOp*) + 0x8080
Module "libhwui.so", in android::uirenderer::renderthread::CanvasContext::getFrameNumber() + 0x21c8
Module "libhwui.so", in android::uirenderer::Stencil::dump() + 0x23c0
Module "libhwui.so", in android::uirenderer::renderthread::RenderProxy::trimMemory(int) + 0x88
Module "libhwui.so", in android::uirenderer::renderthread::RenderProxy::copySurfaceInto(android::sp<android::Surface>&, SkBitmap*) + 0xe4
Module "libhwui.so", in android::uirenderer::renderthread::RenderThread::threadLoop() + 0x98
Module "libutils.so", in android::Thread::_threadLoop(void*) + 0x110
Module "libandroid_runtime.so", in android::AndroidRuntime::javaThreadShell(void*) + 0x74
Module "libc.so", in __pthread_start(void*) + 0xd0
Module "libc.so", in __start_thread + 0x10

dc...@scienceprousa.com <dc...@scienceprousa.com> #29 Nov 1

This issue is still occurring for me on the Pixel XL with the November security update (NDE63V)

ay...@gmail.com <ay...@gmail.com> #30 Nov 1

I see this issue on a Nexus 5X running 7.0 in our App as well. It occurs when the user hits the home button to background the app. 20% of the time it will crash shortly after that, the other 80 crashes when the user returns to the App.

Debug info attached.

deleted
0 B

ta...@gmail.com <ta...@gmail.com> #31 Nov 1

I am seeing this error on Nexus 5x running Nougat as well. It occurs when I start up an intent to capture image, the app then crash with the following stack trace.

E/OpenGLRenderer: Texture is using 0.00B, count = 2 OffscreenBuffer is using 0.00B, count = 0 Layer is using 0.00B, count = 2
A/OpenGLRenderer: Leaked 4 GPU objects!
Fatal signal 6 (SIGABRT), code -6 in tid 15687 (RenderThread)[11-16 23:36:50.997   363: 363 W/] debuggerd: handling request: pid=15657 uid=10147 gid=10147 tid=15687.

This is annoying because it affects production apps and should be addresses as soon as possible with high priority.

nm...@gmail.com <nm...@gmail.com> #32 Nov 1

I have stopped experiencing this issue on my Nexus 5x running build N5D91L (which became available today).

nm...@gmail.com <nm...@gmail.com> #33 Nov 1

[Comment deleted]

[Deleted User] <[Deleted User]> #34 Nov 17

Still happening for me - even with N5D91L.

ay...@gmail.com <ay...@gmail.com> #35 Nov 17

I tried installing N5D91L after seen NManoog reporting no longer experiencing this bug. Initially I believed that the N5D91L update fixed this problem, but after 10 or more cycles of backgrou the app, I experienced the issue again.

After that initial issue appeared, it was much faster to reproduce--i.e. within a single background/foreground cycle.

Here's a copy of the crash info (Nexus 5X, N5D91L):

```
Fatal signal 6 (SIGABRT), code -6 in tid 15612 (RenderThread)
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
'google/bullhead/bullhead:7.0/N5D91L/3425233:user/release-keys'
'rev_1.0'
'arm'
RenderThread  >>> com.myawesomeapp <<<
signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
'Leaked 2 GPU objects!'
    r0 00000000  r1 00003cfc  r2 00000006  r3 00000008
    r4 cc4f4978  r5 00000006  r6 cc4f4920  r7 0000010c
    r8 cc4f4780  r9 00000000  sl ea4607e4  fp ea4607e0
    ip 0000000b  sp cc4f4300  lr eba3e537  pc eba40d94  cpsr 20010010

backtrace:
    #00 pc 00049d94  /system/lib/libc.so (tgkill+12)
    #01 pc 00047533  /system/lib/libc.so (pthread_kill+34)
    #02 pc 0001d885  /system/lib/libc.so (raise+10)
    #03 pc 000193d1  /system/lib/libc.so (__libc_android_abort+34)
    #04 pc 00017014  /system/lib/libc.so (abort+4)
    #05 pc 0000c2c9  /system/lib/libcutils.so (__android_log_assert+112)
    #06 pc 0003ae5f  /system/lib/libhwui.so
    #07 pc 00025893  /system/lib/libhwui.so
    #08 pc 00023eb9  /system/lib/libhwui.so
    #09 pc 00026841  /system/lib/libhwui.so
    #10 pc 000270db  /system/lib/libhwui.so
    #11 pc 00028105  /system/lib/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+80)
    #12 pc 0000e349  /system/lib/libutils.so (_ZN7android6Thread11_threadLoopEPv+140)
    #13 pc 00063fc1  /system/lib/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+80)
    #14 pc 00047003  /system/lib/libc.so (_ZL15__pthread_startPv+22)
    #15 pc 00019e1d  /system/lib/libc.so (__start_thread+6)
```

de...@gmail.com <de...@gmail.com> #36 Nov 18

I have researched a bit, and found where SIGABRT is fired.

http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/GpuMemoryTracker.cpp

There is counter of GpuMemoryTracker instances, and when onGLContextDestroyed() called, it checks that all instances was deleted.
void GpuMemoryTracker::onGLContextDestroyed()
> LOG_ALWAYS_FATAL("Leaked %zd GPU objects!", gObjectSet.size());

gObjectSet is our counter, it is modified in next methods, that are called in constructor and destructor respectively:
void GpuMemoryTracker::startTrackingObject()
void GpuMemoryTracker::stopTrackingObject()

Classes that are ancestors from GpuMemoryTracker are:
class Texture : public GpuMemoryTracker {
class Layer : public VirtualLightRefBase, GpuMemoryTracker {
class OffscreenBuffer : GpuMemoryTracker {

GpuMemoryTracker::onGLContextDestroyed() is static and it is called from RenderState::onGLContextDestroyed() http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderst
And it is called from EglManager::destroy() method http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/EglManager.cpp

Now we should get real leak cause.

nm...@gmail.com <nm...@gmail.com> #37 Nov 19

[Comment deleted]

nm...@gmail.com <nm...@gmail.com> #38 Nov 19

I'd like to retract #31, I'm seeing this issue on N5D91L again.

bd...@gmail.com <bd...@gmail.com> #39 Nov 25

Same issue: build NBD90X - Nexus 5X

- app uses Google Play Service 10.0.0
- maps, location, places

bd...@gmail.com <bd...@gmail.com> #40 Nov 25

android:glEsVersion="0x00020000"

ga...@gmail.com <ga...@gmail.com> #41 Nov 26

Seeing this on my Google Pixel XL.

Happened 10 minutes after using my app and playing around with others.

Honestly this is a pretty serious bug which ruins user experience. Can we please have some word on an updated progress. Thanks.

*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
Build fingerprint: 'google/marlin/marlin:7.1/NDE63V/3389651:user/release-keys'
Revision: '0'
ABI: 'arm64'
pid: 32006, tid: 32037, name: RenderThread >>> com.xxx.xxx <<<
signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
Abort message: 'Leaked 18 GPU objects!'
x0 0000000000000000 x1 0000000000007d25 x2 0000000000000006 x3 0000000000000008
x4 00006e6174736e69 x5 0000000000000000 x6 0000007426ba1000 x7 0000000000000000
x8 0000000000000083 x9 ffffffffffffffdf x10 0000000000000000 x11 0000000000000001
x12 ffffffffffffffff x13 0000000000000000 x14 0000000000000000 x15 000aae9a4fd346f8
x16 00000074257b6ee0 x17 0000007425760250 x18 0000000000000048 x19 00000074081d14f8
x20 0000000000000006 x21 00000074081d1450 x22 000000000000000b x23 00000074232d3050
x24 00000073f9f71f50 x25 00000074160b53d8 x26 00000074160cd540 x27 00000074160b53f0
x28 7fffffffffffffff x29 00000074081d0af0 x30 000000742575d6f8
sp 00000074081d0ad0 pc 0000007425760258 pstate 0000000060000000

backtrace:
#00 pc 000000000006b258 /system/lib64/libc.so (tgkill+8)
#01 pc 00000000000686f4 /system/lib64/libc.so (pthread_kill+64)
#02 pc 0000000000023ce8 /system/lib64/libc.so (raise+24)
#03 pc 000000000001c76c /system/lib64/libc.so (abort+52)
#04 pc 0000000000010bd0 /system/lib64/libcutils.so (__android_log_assert+232)
#05 pc 0000000000055cec /system/lib64/libhwui.so
#06 pc 0000000000035fa0 /system/lib64/libhwui.so
#07 pc 0000000000033604 /system/lib64/libhwui.so
#08 pc 0000000000038174 /system/lib64/libhwui.so
#09 pc 000000000003948c /system/lib64/libhwui.so
#10 pc 000000000003a8cc /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+148)
#11 pc 0000000000012430 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
#12 pc 000000000009e310 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
#13 pc 0000000000067efc /system/lib64/libc.so (_ZL15__pthread_startPv+196)
#14 pc 000000000001d980 /system/lib64/libc.so (__start_thread+16)

ca...@thewelcomepeople.com <ca...@thewelcomepeople.com> #42 Nov 27

Pretty much the same issue as you guys - had to re-flash 22 Nexus 5x phones back to Android 6.0.1 because this broke our app on it!

nm...@gmail.com <nm...@gmail.com> #43 Nov 27

Agreed #40 — I'm also seeing this crash on Google Play Newsstand. I'd imagine that this issue is affecting hundreds of thousands of people at this point. I'm surprised that it isn't getting mo

nm...@gmail.com <nm...@gmail.com> #44 Dec 16

Any updates on this? Even a workaround would be helpful at this point.

dc...@scienceprousa.com <dc...@scienceprousa.com> #45 Dec 16

I can't say for certain, but I feel like I haven't noticed this crash with our app after my Pixel was updated to NMF26O, when it used to happen all the time. Has anyone else noticed a behavior c

ku...@gmail.com <ku...@gmail.com> #46 Dec 16

#44 I think my Nexus 5X started behaving better after NMF26F (Android 7.1.1) update. At least LinkedIn app used to crash a lot before this.

[Deleted User] <[Deleted User]> #47 Dec 16

#43 I made a (bad) workaround
I add.commit() my SupportMapFragment onResume and remove.commit() it onPause..

ju...@gmail.com <ju...@gmail.com> #48 Jan 7

I am getting this Issue in Nexus 6p with 7.1.1 android :

01-06 17:28:26.067 25300-25315/com.visa.cbp.test.app A/libc: Fatal signal 6 (SIGABRT), code -6 in tid 25315 (roidJUnitRunner)

[01-06 17:28:26.068 376: 376 W/]
debuggerd: handling request: pid=25300 uid=10547 gid=10547 tid=25315
01-06 17:28:26.150 25361-25361/? A/DEBUG: *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
01-06 17:28:26.151 25361-25361/? A/DEBUG: Build fingerprint: 'google/angler/angler:7.1.1/NMF26F/3425388:user/release-keys'
01-06 17:28:26.151 25361-25361/? A/DEBUG: Revision: '0'
01-06 17:28:26.151 25361-25361/? A/DEBUG: ABI: 'arm64'
01-06 17:28:26.151 25361-25361/? A/DEBUG: pid: 25300, tid: 25315, name: roidJUnitRunner >>> com.visa.cbp.test.app <<<
01-06 17:28:26.151 25361-25361/? A/DEBUG: signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
01-06 17:28:26.157 25361-25361/? A/DEBUG: Abort message: 'Assertion failed: editable == NULL'
01-06 17:28:26.157 25361-25361/? A/DEBUG: x0 0000000000000000 x1 00000000000062e3 x2 0000000000000006 x3 0000000000000008
01-06 17:28:26.157 25361-25361/? A/DEBUG: x4 0000000000000000 x5 0000000000000000 x6 00000071244ff000 x7 0000000000000000
01-06 17:28:26.157 25361-25361/? A/DEBUG: x8 0000000000000083 x9 fffffffffffffdf x10 0000000000000000 x11 0000000000000001
01-06 17:28:26.157 25361-25361/? A/DEBUG: x12 0000000000000018 x13 0000000000000000 x14 0000000000000000 x15 00040354d634dd25
01-06 17:28:26.157 25361-25361/? A/DEBUG: x16 000000712112aee0 x17 00000071210d4aac x18 0000000000000000 x19 000000711f10f4f8
01-06 17:28:26.157 25361-25361/? A/DEBUG: x20 0000000000000006 x21 000000711f10f450 x22 000000000000000c x23 00000071206e9630
01-06 17:28:26.157 25361-25361/? A/DEBUG: x24 0000000000000008 x25 533581d8387bd2c7 x26 00000071162b1698 x27 533581d8387bd2c7
01-06 17:28:26.157 25361-25361/? A/DEBUG: x28 0000000000000003 x29 000000711f106bc0 x30 00000071210d1ed8
01-06 17:28:26.157 25361-25361/? A/DEBUG: sp 000000711f106ba0 pc 00000071210d4ab4 pstate 0000000060000000
01-06 17:28:26.769 25361-25361/? A/DEBUG: backtrace:
01-06 17:28:26.769 25361-25361/? A/DEBUG: #00 pc 000000000006bab4 /system/lib64/libc.so (tgkill+8)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #01 pc 0000000000068ed4 /system/lib64/libc.so (pthread_kill+64)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #02 pc 0000000000023f58 /system/lib64/libc.so (raise+24)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #03 pc 000000000001c810 /system/lib64/libc.so (abort+52)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #04 pc 0000000000010c24 /system/lib64/libcutils.so (__android_log_assert+224)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #05 pc 0000000000013ea4 /system/lib64/libutils.so (_ZN7android10VectorImpl13editArrayImplEv+256)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #06 pc 0000000000014cf0 /system/lib64/libutils.so (_ZN7android10VectorImpl16editItemLocationEm+56)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #07 pc 0000000000071838 /system/lib64/libbinder.so (_ZN7android12ProcessState23getStrongProxyForHandleEi+152)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #08 pc 000000000005c8c4 /system/lib64/libbinder.so
(_ZN7android16unflatten_binderERKNS_2spINS_12ProcessStateEEERKNS_6ParcelEPNS0_INS_7IBinderEEE+192)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #09 pc 00000000000631d0 /system/lib64/libbinder.so (_ZNK7android6Parcel16readStrongBinderEv+68)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #10 pc 00000000000f0704 /system/lib64/libandroid_runtime.so
01-06 17:28:26.769 25361-25361/? A/DEBUG: #11 pc 0000000001f2a354 /system/framework/arm64/boot-framework.oat (offset 0x1686000) (android.os.Parcel.nativeReadStrongBinder
01-06 17:28:26.769 25361-25361/? A/DEBUG: #12 pc 0000000001f31988 /system/framework/arm64/boot-framework.oat (offset 0x1686000) (android.os.Parcel.readStrongBinder+52)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #13 pc 0000000001f4aac0 /system/framework/arm64/boot-framework.oat (offset 0x1686000) (android.os.ServiceManagerProxy.getServic
01-06 17:28:26.769 25361-25361/? A/DEBUG: #14 pc 0000000001f49d94 /system/framework/arm64/boot-framework.oat (offset 0x1686000) (android.os.ServiceManager.getService+17
01-06 17:28:26.769 25361-25361/? A/DEBUG: #15 pc 000000000202d7ec /system/framework/arm64/boot-framework.oat (offset 0x1686000) (android.security.KeyStore.getInstance+56)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #16 pc 0000000002041468 /system/framework/arm64/boot-framework.oat (offset 0x1686000) (android.security.keystore.AndroidKeyStore
01-06 17:28:26.769 25361-25361/? A/DEBUG: #17 pc 00000000006a8b24 /system/framework/arm64/boot.oat (offset 0x54d000) (java.security.KeyStore.load+64)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #18 pc 00000000000d2734 /system/lib64/libart.so (art_quick_invoke_stub+580)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #19 pc 00000000000df400 /system/lib64/libart.so (_ZN3art9ArtMethod6InvokeEPNS_6ThreadEPjjPNS_6JValueEPKc+208)
01-06 17:28:26.769 25361-25361/? A/DEBUG: #20 pc 000000000029078c /system/lib64/libart.so
(_ZN3art11interpreter34ArtInterpreterToCompiledCodeBridgeEPNS_6ThreadEPNS_9ArtMethodEPKNS_7DexFile8CodeItemEPNS_11ShadowFrameEPNS_6JValueE+312)
01-06 17:28:26.770 25361-25361/? A/DEBUG: #21 pc 0000000000289778 /system/lib64/libart.so
(_ZN3art11interpreter6DoCallILb0ELb0EEEbPNS_9ArtMethodEPNS_6ThreadERNS_11ShadowFrameEPKNS_11InstructionEtPNS_6JValueE+596)
01-06 17:28:26.770 25361-25361/? A/DEBUG: #22 pc 000000000055c664 /system/lib64/libart.so (MterpInvokeVirtualQuick+452)
01-06 17:28:26.770 25361-25361/? A/DEBUG: #23 pc 00000000000c8e94 /system/lib64/libart.so (ExecuteMterpImpl+29972)

---

be...@gmail.com <be...@gmail.com> #49 Feb 3

Any development on this issue? This is still affecting us on Android 7.1.1. The issue has been found in October 2016.. we're in February 2017!

---

ma...@gmail.com <ma...@gmail.com> #50 Mar 27

Getting a lot of these lately:

*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
UUID: x
Build fingerprint: 'Sony/E5823/E5823:7.0/32.3.A.0.376/1066931955:user/release-keys'
Revision: '0'
ABI: 'arm64'
pid: 32120, tid: 32281, name: RenderThread >>> com.example.app <<<
signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
Abort message: 'Leaked 2 GPU objects!'
x0 0000000000000000 x1 00000000000007e19 x2 0000000000000006 x3 0000000000000008
x4 00006e6174736e69 x5 0000000000000000 x6 0000007fa4cf6000 x7 0000000000000000
x8 0000000000000083 x9 fffffffffffffdf x10 0000000000000000 x11 0000000000000001
x12 ffffffffffffffff x13 0000000000000000 x14 0000000000000000 x15 000e28698ddc26b4
x16 0000007fa3720ec8 x17 0000007fa36ce854 x18 0000000000000048 x19 0000007f77c784f8
x20 0000000000000006 x21 0000007f77c78450 x22 0000000000000004 x23 0000007fa46ee030
x24 0000007f74eb6230 x25 0000007f961c6720 x26 0000007f74e1b590 x27 0000007f961c6738
x28 7fffffffffffffff x29 0000007f77c77af0 x30 0000007fa36cbce4
sp 0000007f77c77ad0 pc 0000007fa36ce85c pstate 0000000060000000

backtrace:
  #00 pc 0000000000006b85c /system/lib64/libc.so (tgkill+8)
  #01 pc 0000000000068ce0 /system/lib64/libc.so (pthread_kill+64)
  #02 pc 0000000000024210 /system/lib64/libc.so (raise+24)
  #03 pc 000000000001cc2c /system/lib64/libc.so (abort+52)
  #04 pc 0000000000010bc8 /system/lib64/libcutils.so (__android_log_assert+232)
  #05 pc 0000000000054dbc /system/lib64/libhwui.so
  #06 pc 00000000000351e4 /system/lib64/libhwui.so
  #07 pc 0000000000032858 /system/lib64/libhwui.so
  #08 pc 000000000003751c /system/lib64/libhwui.so
  #09 pc 0000000000038640 /system/lib64/libhwui.so
  #10 pc 0000000000039a98 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+148)
  #11 pc 0000000000012430 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
  #12 pc 000000000009e708 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
  #13 pc 00000000000684e8 /system/lib64/libc.so (_ZL15__pthread_startPv+196)
  #14 pc 000000000001de70 /system/lib64/libc.so (__start_thread+16)

Almost exclusively on Xperia devices:
Xperia XZ (F8331) 10 43,5 %
Xperia Z5 Compact (E5823) 8 34,8 %
Xperia X (F5121) 2 8,7 %
Xperia Z5 (E6653) 1 4,3 %
Xperia X Compact (F5321) 1 4,3 %
Pixel (sailfish) 1 4,3 %

jo...@gmail.com <jo...@gmail.com> #51 Mar 2

Also getting a lot of these same crash reports from users running Android 7.1 on Xperia XZ (F8332) devices, the app in question uses Google Maps views.

*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
UUID: 2c61227e-8523-491d-a5f3-fe04fc9d3935
Build fingerprint: 'Sony/F8332/F8332:7.0/39.2.A.0.374/849105555:user/release-keys'
Revision: '0'
ABI: 'arm64'
pid: 3060, tid: 3114, name: RenderThread >>> com.app.redacted <<<
signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
Abort message: 'Leaked 2 GPU objects!'
  x0 0000000000000000 x1 00000000000000c2a x2 0000000000000006 x3 0000000000000008
  x4 00006e6174736e69 x5 0000000000000000 x6 0000007f93850000 x7 0000000000000000
  x8 0000000000000083 x9 ffffffffffffffdf x10 0000000000000000 x11 0000000000000001
  x12 ffffffffffffffff x13 0000000000000000 x14 0000000000000000 x15 0014bc31a294d71f
  x16 0000007f90084ed0 x17 0000007f9002e9fc x18 0000007f6f5c5180 x19 0000007f747ff4f8
  x20 0000000000000006 x21 0000007f747ff450 x22 000000000000003e x23 0000007f9311c030
  x24 0000007f700cd620 x25 0000007f81d06120 x26 0000007f73fac940 x27 0000007f81d06138
  x28 7fffffffffffffff x29 0000007f747feaf0 x30 0000007f9002be8c
  sp 0000007f747fead0 pc 0000007f9002ea04 pstate 0000000060000000

backtrace:
  #00 pc 0000000000006ba04 /system/lib64/libc.so (tgkill+8)
  #01 pc 0000000000068e88 /system/lib64/libc.so (pthread_kill+64)
  #02 pc 00000000000243b8 /system/lib64/libc.so (raise+24)
  #03 pc 000000000001cdd4 /system/lib64/libc.so (abort+52)
  #04 pc 000000000001085c /system/lib64/libcutils.so (__android_log_assert+232)
  #05 pc 0000000000054dbc /system/lib64/libhwui.so
  #06 pc 00000000000351e4 /system/lib64/libhwui.so
  #07 pc 0000000000032858 /system/lib64/libhwui.so
  #08 pc 000000000003751c /system/lib64/libhwui.so
  #09 pc 0000000000038640 /system/lib64/libhwui.so
  #10 pc 0000000000039a98 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+148)
  #11 pc 0000000000012430 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
  #12 pc 000000000009cf08 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
  #13 pc 0000000000068690 /system/lib64/libc.so (_ZL15__pthread_startPv+196)
  #14 pc 000000000001e018 /system/lib64/libc.so (__start_thread+16)

[Deleted User] <[Deleted User]> #52 Mar 2

[Comment deleted]

bh...@gmail.com <bh...@gmail.com> #53 Mar 2

App runs fine in S7 Edge while crashes on Nexus 6P , both running on Android 7.0.
This happens when I open a particular page...the view model makes asynchronous calls to fetch data from an API..
Stack trace is as:
****

Attempting native Android stacktrace:
    Could not unwind with `libunwind.so`: dlopen failed: library "/data/app/com.myapp365.app-1/lib/arm/libunwind.so" not found
    Could not unwind with `libcorkscrew.so`: dlopen failed: library "/data/app/com.myapp365.app-1/lib/arm/libcorkscrew.so" not found
    No options left to get a native stacktrace :-(

The following assembly referenced from Java.Interop.dll could not be loaded:
Assembly: System.Runtime (assemblyref_index=1)
Version: 4.0.0.0
Public Key: b03f5f7f11d50a3a
The assembly was not found in the Global Assembly Cache, a path listed in the MONO_PATH environment variable, or in the location of the executing assembly (/).
Failed to load assembly Java.Interop[0xeaa5e020]

Got a SIGSEGV while executing native code. This usually indicates
a fatal error in the mono runtime or one of the native libraries
used by your application.

libc (2899): Fatal signal 6 (SIGABRT), code -6 in tid 2926 (RenderThread)

---

pe...@mappau.com <pe...@mappau.com> #54 Apr 4

App runs fine on Nexus, Samsung but crashes randomly on Sony Xperia Compact

I think it's related to Google MapView

A/libc: Fatal signal 11 (SIGSEGV), code 1, fault addr 0x7f5ddebba0 in tid 7376 (FinalizerDaemon)

[04-04 11:41:43.953 466: 466 W/]
debuggerd: handling request: pid=7367 uid=10212 gid=10212 tid=7376
E/LocSvc_eng: E/Calling gnss_sv_status_cb
D/clmlib: Got activities:0x0000000E
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
UUID: a2971786-248f-4cc1-9282-cc6150fadf7b
Build fingerprint: 'Sony/F5321/F5321:7.0/34.2.A.0.311/2823159678:user/release-keys'
Revision: '0'
ABI: 'arm64'
pid: 7367, tid: 7376, name: FinalizerDaemon >>> com.***.android <<<
signal 11 (SIGSEGV), code 1 (SEGV_MAPERR), fault addr 0x7f5ddebba0
x0 0000007f5ddebb50 x1 0000007f8a40dcb4 x2 0000007f5ddebb50 x3 0000000000000000
x4 000000006fa1414c x5 0000000000000000 x6 0000000072f25b78 x7 0000007f61a2b4e0
x8 0000000000000000 x9 0000000000000000 x10 0000000000430000 x11 0000000000000000
x12 0000007f8a40dae0 x13 0000007f8b2fb000 x14 0000007f8a40dbf8 x15 0000000000000000
x16 0000007f8c1b74e8 x17 0000007f8c181d60 x18 00000000714b30fc x19 0000007f5ddebb50
x20 0000007f8c4e237c x21 00000000134fefc0 x22 00000000130e3fd8 x23 00000000134fefc0
x24 0000000000001196 x25 00000000130e3fd8 x26 0000007f7d44fe98 x27 d515bc5f27d015d4
x28 0000007f8b1fe000 x29 0000007f8a40dc70 x30 0000007f8c4e2398
sp 0000007f8a40dc60 pc 0000007f8c181d70 pstate 0000000020000000
backtrace:
#00 pc 000000000005cd70 /system/lib64/libbinder.so (_ZN7android6Parcel14freeDataNoInitEv+16)
#01 pc 00000000000ee394 /system/lib64/libandroid_runtime.so
#02 pc 0000000001f83644 /system/framework/arm64/boot-framework.oat (offset 0x16de000) (android.os.Parcel.nativeDestroy+128)
#03 pc 0000000001f82770 /system/framework/arm64/boot-framework.oat (offset 0x16de000) (android.os.Parcel.destroy+76)
#04 pc 0000000001f87e5c /system/framework/arm64/boot-framework.oat (offset 0x16de000) (android.os.Parcel.finalize+40)
#05 pc 0000000000689510 /system/framework/arm64/boot-core-libart.oat (offset 0x4a0000) (java.lang.Daemons$FinalizerDaemon.doFinalize+140)
#06 pc 00000000006897ec /system/framework/arm64/boot-core-libart.oat (offset 0x4a0000) (java.lang.Daemons$FinalizerDaemon.run+520)
#07 pc 000000000065ea70 /system/framework/arm64/boot.oat (offset 0x5a1000) (java.lang.Thread.run+60)
#08 pc 00000000000d1ab4 /system/lib64/libart.so (art_quick_invoke_stub+580)
#09 pc 00000000000de780 /system/lib64/libart.so (_ZN3art9ArtMethod6InvokeEPNS_6ThreadEPjjPNS_6JValueEPKc+204)
#10 pc 00000000004265f0 /system/lib64/libart.so (_ZN3artL18InvokeWithArgArrayERKNS_33ScopedObjectAccessAlreadyRunnableEPNS_9ArtMethodEPNS_8ArgArrayEPNS_6JValueEP
#11 pc 00000000004278f8 /system/lib64/libart.so (_ZN3art35InvokeVirtualOrInterfaceWithJValuesERKNS_33ScopedObjectAccessAlreadyRunnableEP8_jobjectP10_jmethodIDP6jvalue+
#12 pc 0000000000447b38 /system/lib64/libart.so (_ZN3art6Thread14CreateCallbackEPv+1100)
#13 pc 00000000000685bc /system/lib64/libc.so (_ZL15__pthread_startPv+196)
#14 pc 000000000001deb0 /system/lib64/libc.so (__start_thread+16)

---

r....@gmail.com <r....@gmail.com> #55 Apr 1

Same issue here. We're using the camera with a TextureView and it's crashing when leaving the screen.

---

ds...@gmail.com <ds...@gmail.com> #56 Apr 2

+1

---

gr...@gmail.com <gr...@gmail.com> #57 Apr 2

Same problem in Sony Xperia Z5 compact with android 7.0

---

rr...@gmail.com <rr...@gmail.com> #58 May 1

E/OpenGLRenderer: Texture is using 0.00B, count = 1
OffscreenBuffer is using 0.00B, count = 0

Layer is using 0.00B, count = 1
05-14 19:29:30.911 31706-31756/com.package.test A/OpenGLRenderer: Leaked 2 GPU objects!
--------- beginning of crash

05-14 19:29:30.912 31706-31756/com.package.test A/libc: Fatal signal 6 (SIGABRT), code -6 in tid 31756 (RenderThread)

[05-14 19:29:30.919 10376:10376 W/]

debuggerd: handling request: pid=31706 uid=10371 gid=10371 tid=31756

This is my log using Xperia X Compact
This thread isn't closed yet. Please google solve it...

[Deleted User] <[Deleted User]> #59 Jun 9

Same problem. Having crash report on following Android 7.0 above device:
P9 Plus (HWVIE)
Mate 9 (HWMHA)
Galaxy S6 Edge (zerolte)
P10 Plus (HWVKY)
Galaxy S8 (dreamlte)
Galaxy S8+ (dream2lteks)
Galaxy S8+ (dream2lte)

backtrace:
native: pc 0000000000006b598 /system/lib64/libc.so (tgkill+8)
native: pc 0000000000068a34 /system/lib64/libc.so (pthread_kill+64)
native: pc 0000000000023de8 /system/lib64/libc.so (raise+24)
native: pc 0000000000001c86c /system/lib64/libc.so (abort+52)
native: pc 00000000000125fc /vendor/lib64/libcutils.so (__android_log_assert+232)
native: pc 0000000000032d0c /system/lib64/libhwui.so
native: pc 0000000000033440 /system/lib64/libhwui.so
native: pc 000000000003a9b0 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread22dispatchFrameCallbacksEv+212)
native: pc 000000000003ab94 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+156)
native: pc 000000000001249c /vendor/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
native: pc 00000000000a3f00 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
native: pc 000000000006823c /system/lib64/libc.so (_ZL15__pthread_startPv+196)
native: pc 0000000000001da80 /system/lib64/libc.so (__start_thread+16)

---

va...@gmail.com <va...@gmail.com> #60 Jul 1(

Same problem just on Samsung A5 under Android (7.0).
Also have tested on Nexus 5X (7.1.2), Nexus 5 (7.1.2), Samsung Galaxy S7(7.0) & a lot of other devices on Android < 7 but all works fine.

[Deleted User] <[Deleted User]> #61 Jul 18

```
native: pc 000000000004a230 /system/lib/libc.so (tgkill+12)
native: pc 00000000000479c3 /system/lib/libc.so (pthread_kill+34)
native: pc 000000000001d9c5 /system/lib/libc.so (raise+10)
native: pc 0000000000019511 /system/lib/libc.so (__libc_android_abort+34)
native: pc 0000000000017150 /system/lib/libc.so (abort+4)
native: pc 000000000000c687 /system/lib/libcutils.so (__android_log_assert+114)
native: pc 000000000003cf15 /system/lib/libhwui.so
native: pc 0000000000027183 /system/lib/libhwui.so
native: pc 0000000000025761 /system/lib/libhwui.so
native: pc 00000000000281d1 /system/lib/libhwui.so
native: pc 0000000000028a73 /system/lib/libhwui.so
native: pc 0000000000029aa1 /system/lib/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+80)
native: pc 000000000000e329 /system/lib/libutils.so (_ZN7android6Thread11_threadLoopEPv+144)
native: pc 000000000006aecd /system/lib/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+80)
native: pc 0000000000047493 /system/lib/libc.so (_ZL15__pthread_startPv+22)
native: pc 0000000000019f6d /system/lib/libc.so (__start_thread+6)
```

Any updates to this? Like others, I'm getting this issue when stopping a fragment that's using Camera2 Api. I can look into making a sample project if need be. Thanks.

---

tj...@tiqets.com <tj...@tiqets.com> #62 Jul 19

I'm getting this now on all Android 7.0 phones: Galaxy s6. s7 amd s8.
I open the app, with a GoogleMap in a Fragment in a ViewPager, then i close the app: Crash: Leaked 2 gpu objects.

This is happening A LOT now

/audit: type=1400 audit(1500463914.790:303): avc: denied { search } for pid=26159 comm="debuggerd64" name="com.google.android.gms" dev="sda18" ino=457251 scontext=u:r:debugge
tcontext=u:object_r:app_data_file:s0:c512,c768 tclass=dir permissive=0 SEPF_SECMOBILE_7.0_0005
07-19 13:31:54.808 3017-3017/? E/audit: type=1400 audit(1500463914.800:304): avc: denied { search } for pid=26159 comm="debuggerd64" name="com.google.android.gms" dev="sda18"
scontext=u:r:debuggerd:s0 tcontext=u:object_r:app_data_file:s0:c512,c768 tclass=dir permissive=0 SEPF_SECMOBILE_7.0_0005
07-19 13:31:54.888 26159-26159/? A/DEBUG: *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
07-19 13:31:54.889 26159-26159/? A/DEBUG: Build fingerprint: 'samsung/zerofltexx/zeroflte:7.0/NRD90M/G920FXXU5EQFC:user/release-keys'
07-19 13:31:54.889 26159-26159/? A/DEBUG: Revision: '11'
07-19 13:31:54.889 26159-26159/? A/DEBUG: ABI: 'arm64'
07-19 13:31:54.889 26159-26159/? A/DEBUG: pid: 24317, tid: 26072, name: RenderThread >>> com.tiqets.tiqetsapp <<<
07-19 13:31:54.889 26159-26159/? A/DEBUG: signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
07-19 13:31:54.898 26159-26159/? A/DEBUG: Abort message: 'Leaked 2 GPU objects!'

```
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x0  0000000000000000  x1  00000000000065d8  x2  0000000000000006  x3  0000000000000008
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x4  0000000000000021  x5  0000000000000000  x6  000000781b39c000  x7  0000000000000000
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x8  0000000000000083  x9  ffffffffffffffdf  x10 0000000000000000  x11 0000000000000001
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x12 ffffffffffffffff  x13 0000000000000000  x14 0000000000000000  x15 002d9814699b040d
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x16 0000007818d3dee0  x17 0000007818ce75ac  x18 0000000000000000  x19 00000077e5a224f8
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x20 0000000000000006  x21 00000077e5a22450  x22 000000000000001d  x23 0000007817121008
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x24 00000077d36fdc40  x25 00000000000fd000  x26 9b60690fa71e0328  x27 00000077ea0178b8
07-19 13:31:54.898 26159-26159/? A/DEBUG:    x28 7fffffffffffffff  x29 00000077e5a21af0  x30 0000007818ce4a54
07-19 13:31:54.898 26159-26159/? A/DEBUG:    sp  00000077e5a21ad0  pc  0000007818ce75b4  pstate 0000000060000000
07-19 13:31:54.907 26159-26159/? A/DEBUG: backtrace:
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #00 pc 000000000006b5b4  /system/lib64/libc.so (tgkill+8)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #01 pc 0000000000068a50  /system/lib64/libc.so (pthread_kill+64)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #02 pc 000000000023f68  /system/lib64/libc.so (raise+24)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #03 pc 000000000001c9ec  /system/lib64/libc.so (abort+52)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #04 pc 00000000000112ec  /system/lib64/libcutils.so (__android_log_assert+232)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #05 pc 0000000000057700  /system/lib64/libhwui.so
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #06 pc 00000000000372f4  /system/lib64/libhwui.so
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #07 pc 0000000000034ad0  /system/lib64/libhwui.so
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #08 pc 0000000000039620  /system/lib64/libhwui.so
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #09 pc 000000000003a720  /system/lib64/libhwui.so
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #10 pc 000000000003bb5c  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+148)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #11 pc 0000000000012430  /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #12 pc 00000000000a4d40  /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #13 pc 0000000000068258  /system/lib64/libc.so (_ZL15__pthread_startPv+196)
07-19 13:31:54.907 26159-26159/? A/DEBUG:    #14 pc 000000000001dc00  /system/lib64/libc.so (__start_thread+16)
07-19 13:31:55.828 26169-26169/? E/Zygote: v2
07-19 13:31:55.828 26169-26169/? E/Zygote: accessInfo : 0
07-19 13:31:55.929 3017-3017/? E/audit: type=1400 audit(1500463915.920:305): avc:  denied  { search } for  pid=26159 comm="debuggerd64" name="com.google.android.gms" dev="sda18"
scontext=u:r:debuggerd:s0 tcontext=u:object_r:app_data_file:s0:c512,c768 tclass=dir permissive=0 SEPF_SECMOBILE_7.0_0005
07-19 13:31:56.655 3096-3096/? E/lowmemorykiller: Error writing /proc/24317/oom_score_adj; errno=22
07-19 13:31:56.677 26197-26197/? E/Zygote: v2
07-19 13:31:56.678 26197-26197/? E/Zygote: accessInfo : 0
07-19 13:31:56.741 3657-3810/? E/InputDispatcher: channel ~ Channel is unrecoverably broken and will be disposed!
07-19 13:31:56.794 26218-26218/? E/Zygote: v2
07-19 13:31:56.794 26218-26218/? E/Zygote: accessInfo : 0
07-19 13:31:57.502 26242-26242/? E/Zygote: v2
07-19 13:31:57.504 26242-26242/? E/Zygote: accessInfo : 0
```

---

co...@blurry-app.com <co...@blurry-app.com> #63 Jul 29

native: pc 000000000007a6c4 /system/lib64/libc.so (tgkill+8)

native: pc 0000000000077920 /system/lib64/libc.so (pthread_kill+64)

native: pc 000000000002538c /system/lib64/libc.so (raise+24)

native: pc 000000000001d24c /system/lib64/libc.so (abort+52)

native: pc 000000000001225c /system/lib64/libcutils.so (__android_log_assert+224)

native: pc 00000000000610e0 /system/lib64/libhwui.so

native: pc 000000000003908c /system/lib64/libhwui.so

native: pc 000000000003609c /system/lib64/libhwui.so

native: pc 000000000003b4fc /system/lib64/libhwui.so

native: pc 000000000003c520 /system/lib64/libhwui.so

native: pc 000000000003e694 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+152)

native: pc 00000000000127f0 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+336)

native: pc 00000000000a50b0 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)

native: pc 00000000000770f4 /system/lib64/libc.so (_ZL15__pthread_startPv+204)

native: pc 000000000001e7d0 /system/lib64/libc.so (__start_thread+16)

---

js...@gmail.com <js...@gmail.com> #64 Jul 3

Over the last 2 months, we have over 30k of these tgkill crashes effecting 14k+ users. I have spent the last couple of weeks slowly removing any 3rd party libraries we're using and releasing s
if I can track down what is causing these crashes. We released a staged rollout with only removing 'com.google.android.gms:play-services-maps:11.0.1'. After watching it all weekend, there v
the tgkill crash. Yes, this issue is caused by maps!

native: pc 000000000007a6c4 /system/lib64/libc.so (tgkill+8)
native: pc 0000000000077920 /system/lib64/libc.so (pthread_kill+64)
native: pc 000000000002538c /system/lib64/libc.so (raise+24)
native: pc 000000000001d24c /system/lib64/libc.so (abort+52)
native: pc 00000000000122b0 /system/lib64/libcutils.so (__android_log_assert+224)
native: pc 00000000000610e0 /system/lib64/libhwui.so
native: pc 000000000003908c /system/lib64/libhwui.so
native: pc 000000000003609c /system/lib64/libhwui.so
native: pc 000000000003b4fc /system/lib64/libhwui.so

native: pc 000000000003c520 /system/lib64/libhwui.so
native: pc 000000000003e694 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+152)
native: pc 00000000000127f0 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+336)
native: pc 00000000000a50b0 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
native: pc 00000000000770f4 /system/lib64/libc.so (_ZL15__pthread_startPv+204)
native: pc 000000000001e7d0 /system/lib64/libc.so (__start_thread+16)

ap...@gmail.com <ap...@gmail.com> #65 Aug

We are facing this issue as well, with thousands of users effected daily. So far all research points to maps.

For more info: https://stackoverflow.com/questions/44080809/android-7-native-crash-libc-so-tgkill

[Deleted User] <[Deleted User]> #66 Aug

We have the same issue as well and it affects thousands of users.
We are using a ViewPager that contains a Fragment with a MapView.
The crash seems to occurs when our app is in the background and other apps are started.

Affected devices:
Galaxy S6 (zeroflte)
Xperia Z5 Compact (E5823)
Galaxy S6 Edge (zerolte)
Galaxy S8 (dreamlte)
Galaxy S8+ (dream2lte)
Xperia X (F5121)
Xperia X Compact (F5321)
Xperia Z5 (E6653)
Xperia Z5 Premium (E6853)
Xperia Z3+ (E6553)
Xperia XZ (F8331)
Xperia Z5 dual (E6633)
Xperia X Performance (F8131)
Nexus 5X (bullhead)
Nexus 6P (angler)
Galaxy S6 Edge+ (zenlte)
Galaxy S6 (zeroltektt)
Honor 8 (HWFRD)
Xperia Z5 Premium (E6883)
Xperia Z5 Premium Dual (E6833)

Stacktrace:
native: pc 000000000006b930 /system/lib64/libc.so (tgkill+8)
native: pc 0000000000068db4 /system/lib64/libc.so (pthread_kill+64)
native: pc 0000000000024250 /system/lib64/libc.so (raise+24)
native: pc 000000000001cc6c /system/lib64/libc.so (abort+52)
native: pc 0000000000010cfc /system/lib64/libcutils.so (__android_log_assert+232)
native: pc 0000000000054dbc /system/lib64/libhwui.so
native: pc 00000000000351e4 /system/lib64/libhwui.so
native: pc 0000000000032858 /system/lib64/libhwui.so
native: pc 000000000003751c /system/lib64/libhwui.so
native: pc 0000000000038640 /system/lib64/libhwui.so
native: pc 0000000000039a98 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+148)
native: pc 0000000000012430 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
native: pc 000000000009e7f8 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
native: pc 00000000000685bc /system/lib64/libc.so (_ZL15__pthread_startPv+196)
native: pc 000000000001deb0 /system/lib64/libc.so (__start_thread+16)

ks...@gmail.com <ks...@gmail.com> #67 Aug 2

any way to solve it?

[Deleted User] <[Deleted User]> #68 Sep 2

I found one possible cause of this problem.

When an animating view is removed from ViewGroup, removeViewInternal() postpones call of `dispatchDetachedFromWindow()` until animation finishes.
http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/5.1.1_r1/android/view/ViewGroup.java#4218
However, the view is remove from mChildren instantly.
In case a process receives trimMemory request with TRIM_MEMORY_COMPLETE level, WindowManagerGlobal destroys all hardware resources by traversing view hierarchy from ViewRootIm
version of trimMemory(). http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/WindowManagerGlobal.java#491
C++ trimMemory() eventually calls EglManager.destroy() and eventually comes at http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/GpuMemoryTracker.cpp#78, where crash
Usually WindowManagerGlobal destroys all hardware resources, but when there are a view that is animating and removed, that's resources are not freed, therefore results in crash.

Here is a simple case to reproduce the issue.
You can run `am send-trim-memory <pid> COMPLETE` command in order to simulate low memory situation.

```
public class MainActivity extends AppCompatActivity {
    private static final String TAG = MainActivity.class.getName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final FrameLayout root = (FrameLayout) findViewById(R.id.root);

        final TextureView textureView = new TextureView(this);
        textureView.setSurfaceTextureListener(new TextureView.SurfaceTextureListener() {
            @Override
            public void onSurfaceTextureAvailable(SurfaceTexture surfaceTexture, int i, int i1) {
                Surface surface = new Surface(surfaceTexture);
                Canvas canvas = surface.lockCanvas(null);
                canvas.drawColor(Color.GRAY);
                surface.unlockCanvasAndPost(canvas);
                Log.d(TAG, "onSurfaceTextureAvailable");
            }

            @Override
            public void onSurfaceTextureSizeChanged(SurfaceTexture surfaceTexture, int i, int i1) {
            }

            @Override
            public boolean onSurfaceTextureDestroyed(SurfaceTexture surfaceTexture) {
                Log.d(TAG, "onSurfaceTextureDestroyed");
                return true;
            }

            @Override
            public void onSurfaceTextureUpdated(SurfaceTexture surfaceTexture) {

            }
        });
        root.addView(textureView, new FrameLayout.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT, ViewGroup.LayoutParams.MATCH_PARENT));
        new Handler(Looper.getMainLooper()).postDelayed(new Runnable() {
            @Override
            public void run() {
                root.removeView(textureView);
                Toast.makeText(MainActivity.this, "Detached! Now put app into background and call \"am send-trim-memory " + Process.myPid() + " COMPLETE\" from adb shell.", Toast.LENGTH_
            }
        }, 5000);

        Animation animation = AnimationUtils.loadAnimation(this, R.anim.myanimation);
        animation.setDuration(30 * 1000);
        textureView.startAnimation(animation);
    }
}
```

This is trace I got from the program above.

```
09-21 23:17:36.325 21864-21897/com.example.crashsample E/OpenGLRenderer: Texture is using 0.00B, count = 1
                                                                          OffscreenBuffer is using 0.00B, count = 0
                                                                          Layer is using 0.00B, count = 1
09-21 23:17:36.325 21864-21897/com.example.crashsample A/OpenGLRenderer: Leaked 2 GPU objects!
09-21 23:17:36.325 21864-21897/com.example.crashsample A/libc: Fatal signal 6 (SIGABRT), code -6 in tid 21897 (RenderThread)

                                                                  [ 09-21 23:17:36.326   353:  353 W/        ]
                                                                  debuggerd: handling request: pid=21864 uid=10102 gid=10102 tid=21897
09-21 23:17:36.394 22229-22229/? A/DEBUG: *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
09-21 23:17:36.394 22229-22229/? A/DEBUG: Build fingerprint: 'HUAWEI/WAS-LX2J/HWWAS-H:7.0/HUAWEIWAS-LX2J/C635B106:user/release-keys'
09-21 23:17:36.394 22229-22229/? A/DEBUG: Revision: '0'
09-21 23:17:36.394 22229-22229/? A/DEBUG: ABI: 'arm64'
09-21 23:17:36.394 22229-22229/? A/DEBUG: pid: 21864, tid: 21897, name: RenderThread  >>> com.example.crashsample <<<
09-21 23:17:36.394 22229-22229/? A/DEBUG: signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
09-21 23:17:36.397 22229-22229/? A/DEBUG: Abort message: 'Leaked 2 GPU objects!'
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x0  0000000000000000  x1  0000000000005589  x2  0000000000000006  x3  0000000000000008
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x4  00006e6174736e69  x5  0000000000000000  x6  0000007aae09b000  x7  0000000000000000
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x8  0000000000000083  x9  fffffffffffffdf  x10 0000000000000000  x11 0000000000000001
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x12 ffffffffffffffff  x13 0000000000000000  x14 0000000000000000  x15 0004d922e3920ff1
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x16 0000007aacbcbec8  x17 0000007aacb74c28  x18 0000000000000000  x19 0000007aa1dfd4f8
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x20 0000000000000006  x21 0000007aa1dfd450  x22 0000000000000009  x23 0000007aaac24048
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x24 0000007a8bbda000  x25 0000007aaa08f9f8  x26 7fffffffffffffff  x27 0000000000000000
09-21 23:17:36.397 22229-22229/? A/DEBUG:     x28 431bde82d7b634db  x29 0000007aa1dfcb00  x30 0000007aacb720d0
09-21 23:17:36.397 22229-22229/? A/DEBUG:     sp  0000007aa1dfcae0  pc  0000007aacb74c30  pstate 0000000060000000
09-21 23:17:36.402 22229-22229/? A/DEBUG: backtrace:
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #00 pc 000000000006bc30  /system/lib64/libc.so (tgkill+8)
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #01 pc 00000000000690cc  /system/lib64/libc.so (pthread_kill+64)
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #02 pc 0000000000023e68  /system/lib64/libc.so (raise+24)
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #03 pc 000000000001c8ec  /system/lib64/libc.so (abort+52)
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #04 pc 0000000000012c98  /vendor/lib64/libcutils.so (__android_log_assert+232)
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #05 pc 0000000000057398  /system/lib64/libhwui.so
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #06 pc 0000000000036c28  /system/lib64/libhwui.so
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #07 pc 0000000000033e48  /system/lib64/libhwui.so
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #08 pc 0000000000038f38  /system/lib64/libhwui.so
09-21 23:17:36.402 22229-22229/? A/DEBUG:     #09 pc 000000000003a25c  /system/lib64/libhwui.so
09-21 23:17:36.403 22229-22229/? A/DEBUG:     #10 pc 000000000003b6a0  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+156)
09-21 23:17:36.403 22229-22229/? A/DEBUG:     #11 pc 00000000000125e8  /vendor/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
09-21 23:17:36.403 22229-22229/? A/DEBUG:     #12 pc 00000000000a66a4  /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
09-21 23:17:36.403 22229-22229/? A/DEBUG:     #13 pc 00000000000688d4  /system/lib64/libc.so (_ZL15__pthread_startPv+196)
```

09-21 23:17:36.403 22229-22229/? A/DEBUG: #14 pc 0000000000001db00 /system/lib64/libc.so (__start_thread+16)
09-21 23:17:36.518 1169-1295/? I/EventHub: EventHub monitor: no key events in the past 3868s,

I request android developers to fix such a loophole, but we users can cope with this.

The simplest is to remove all GPU resource users when the activity goes to background.

---

nc...@gmail.com <nc...@gmail.com> #69 Nov 29

This seems to have been fixed in Oreo, because it only seems to affect Nougat devices.

Still, lack of updates, and the popularity of devices such as the Galaxy S8, means this issue is 10x more prevalent than any other crash, and affects something like 0.5% of all sessions.

I'm using a TextureView inside a ViewPager.

Has anyone came up with a workaround?

#68, how do you suggest we should "remove all GPU resource users when the activity goes to background"? How would you fix your own example (the view is already removed from hierarchy

Thanks.

---

jt...@googlemail.com <jt...@googlemail.com> #70 Nov 30

I'm also seeing this stack trace for an app that uses a ViewPager. All affected devices are running Android 6.0.

---

pr...@gmail.com <pr...@gmail.com> #71 Dec 18

Is anyone working on this bug?

I've run into this as well (Samsung Galxy S8, Android 7.0):

*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
Build fingerprint: 'samsung/dreamltexx/dreamlte:7.0/NRD90M/G950FXXU1AQJ1:user/release-keys'
Revision: '10'
ABI: 'arm64'
pid: 31586, tid: 31701, name: RenderThread >>> se.dxapps.skidspar.beta <<<
signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
Abort message: 'Leaked 2 GPU objects!'
  x0 0000000000000000 x1 0000000000007bd5 x2 0000000000000006 x3 0000000000000008
  x4 0000000000000021 x5 0000000000000000 x6 0000007a68c8e000 x7 0000000000000000
  x8 0000000000000083 x9 ffffffffffffffdf x10 0000000000000000 x11 ffffffffffffffff
  x12 0000000000000018 x13 0000000000000000 x14 0000000000000000 x15 001f8013189a298a
  x16 0000007a665a5ed0 x17 0000007a6654d9f4 x18 0000000000000000 x19 0000007a186b24f8
  x20 0000000000000006 x21 0000007a186b2450 x22 000000000000000b x23 0000000000000000
  x24 0000007a657a0008 x25 00000079f9a2f4c0 x26 7fffffffffffffff x27 0000007a038e15a0
  x28 0000007a3d2afbf8 x29 0000007a186b1ad0 x30 0000007a6654ad14
  sp 0000007a186b1ab0 pc 0000007a6654d9fc pstate 0000000060000000

backtrace:
  #00 pc 000000000006f9fc /system/lib64/libc.so (tgkill+8)
  #01 pc 000000000006cd10 /system/lib64/libc.so (pthread_kill+64)
  #02 pc 0000000000025078 /system/lib64/libc.so (raise+24)
  #03 pc 000000000001cc04 /system/lib64/libc.so (abort+52)
  #04 pc 0000000000011c4c /system/lib64/libcutils.so (__android_log_assert+252)
  #05 pc 0000000000059fc4 /system/lib64/libhwui.so
  #06 pc 0000000000037c10 /system/lib64/libhwui.so
  #07 pc 00000000000351b8 /system/lib64/libhwui.so
  #08 pc 000000000003a0e4 /system/lib64/libhwui.so
  #09 pc 000000000003b2b8 /system/lib64/libhwui.so
  #10 pc 000000000003c9f8 /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+152)
  #11 pc 0000000000012930 /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
  #12 pc 00000000000a50b0 /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
  #13 pc 000000000006c4a0 /system/lib64/libc.so (_ZL15__pthread_startPv+208)
  #14 pc 000000000001e220 /system/lib64/libc.so (__start_thread+16)

---

nc...@gmail.com <nc...@gmail.com> #72 Dec 18

I assume the bug has been fixed. It mostly affects Android 7.0 (maybe 7.1.x). I haven't seen it on Android 8.x.
The problem is usage of Oreo is still residual, and usage of Nougat is increasing, not reducing.
A mitigation or workaround would be greatly appreciated.
Alas, whatever we write here, seems to fall on deaf ears.

---

sr...@gmail.com <sr...@gmail.com> #73 Jan 3

I saw it on 8.0, please check following log
12-27 17:48:09.398 10186 26944 26944 F DEBUG   : *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
12-27 17:48:09.398 10186 26944 26944 F DEBUG   : Build fingerprint: 'samsung/dream2qltezc/dream2qltechn:8.0.0/OPR1.170623.032/G9550ZCU2ZQL8:user/release-keys'
12-27 17:48:09.398 10186 26944 26944 F DEBUG   : Revision: '12'
12-27 17:48:09.398 10186 26944 26944 F DEBUG   : ABI: 'arm'
12-27 17:48:09.398 10186 26944 26944 F DEBUG   : pid: 16617, tid: 22593, name: RenderThread >>> com.samsung.android.app.sreminder <<<
12-27 17:48:09.398 10186 26944 26944 F DEBUG   : signal 11 (SIGSEGV), code 1 (SEGV_MAPERR), fault addr 0x0

```
12-27 17:48:09.398 10186 26944 26944 F DEBUG   : Cause: null pointer dereference
12-27 17:48:09.398 10186 26944 26944 F DEBUG   :     r0 000007f0  r1 00001fff  r2 f0d01e00  r3 0000000d
12-27 17:48:09.398 10186 26944 26944 F DEBUG   :     r4 ffffffff  r5 00000000  r6 ccbdc400  r7 00000000
12-27 17:48:09.399 10186 26944 26944 F DEBUG   :     r8 7ffff000  r9 e05ffe08  sl 00000000  fp cb46b764
12-27 17:48:09.399 10186 26944 26944 F DEBUG   :     ip f1cdbd60  sp cb46b3a8  lr f1cb756f  pc f1cb219a  cpsr 200f0030
12-27 17:48:09.506 10186 26944 26944 F DEBUG   : backtrace:
12-27 17:48:09.506 10186 26944 26944 F DEBUG   :     #00 pc 0006719a  /system/lib/libc.so (je_huge_salloc+133)
12-27 17:48:09.506 10186 26944 26944 F DEBUG   :     #01 pc 0006c56b  /system/lib/libc.so (ifree+274)
12-27 17:48:09.506 10186 26944 26944 F DEBUG   :     #02 pc 0006c8d1  /system/lib/libc.so (je_free+72)
12-27 17:48:09.506 10186 26944 26944 F DEBUG   :     #03 pc 0004c467  /system/lib/libhwui.so (_ZN7android10uirenderer11DisplayList16cleanupResourcesEv+194)
12-27 17:48:09.506 10186 26944 26944 F DEBUG   :     #04 pc 0004c33b  /system/lib/libhwui.so (_ZN7android10uirenderer11DisplayListD1Ev+18)
12-27 17:48:09.506 10186 26944 26944 F DEBUG   :     #05 pc 0004c673  /system/lib/libhwui.so (_ZN7android10uirenderer11DisplayListD0Ev+2)
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #06 pc 000670b5  /system/lib/libhwui.so (_ZN7android10uirenderer10RenderNode17deleteDisplayListERNS0_12TreeObserverEPNS0_
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #07 pc 000680db  /system/lib/libhwui.so (_ZN7android10uirenderer10RenderNode15syncDisplayListERNS0_12TreeObserverEPNS0_8
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #08 pc 0006807f  /system/lib/libhwui.so
(_ZN7android10uirenderer10RenderNode29pushStagingDisplayListChangesERNS0_12TreeObserverERNS0_8TreeInfoE+34)
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #09 pc 00067a69  /system/lib/libhwui.so (_ZN7android10uirenderer10RenderNode15prepareTreeImplERNS0_12TreeObserverERNS0_8
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #10 pc 0002df13  /system/lib/libhwui.so
(_ZNKSt3__18functionIFvPN7android10uirenderer10RenderNodeERNS2_12TreeObserverERNS2_8TreeInfoEbEEclES4_S6_S8_b+50)
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #11 pc 0004c855  /system/lib/libhwui.so
(_ZN7android10uirenderer11DisplayList22prepareListAndChildrenERNS0_12TreeObserverERNS0_8TreeInfoEbNSt3__18functionIFvPNS0_10RenderNodeES3_S5_bEEE+72)
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #12 pc 00067aa3  /system/lib/libhwui.so (_ZN7android10uirenderer10RenderNode15prepareTreeImplERNS0_12TreeObserverERNS0_8
12-27 17:48:09.507 10186 26944 26944 F DEBUG   :     #13 pc 0002df13  /system/lib/libhwui.so
(_ZNKSt3__18functionIFvPN7android10uirenderer10RenderNodeERNS2_12TreeObserverERNS2_8TreeInfoEbEEclES4_S6_S8_b+50)
```

---

ds...@gmail.com <ds...@gmail.com> #74 Jan 1

Any update on this? I'd echo #72: "A mitigation or workaround would be greatly appreciated.".
Usage of Android 7 is increasing and seemingly Google Map causes this crash too https://stackoverflow.com/questions/44080809/android-7-native-crash-libc-so-tgkill

---

[Deleted User] <[Deleted User]> #75 Feb

I think I fixed this issue for my case, at least I can't reproduce it anymore.

In one of the crashes console showed following message:
"Below Textures or Layers was not cleared even though Application being closed.(It should be fixed by Application side)"

In the onStop method for Fragment/Activity I do following things:
- call clear() on instance of GoogleMap
- set all listeners to null on instance of GoogleMap
- set all adapters to null on instance of GoogleMap
- call removeAllViews() on instance of MapView

When Fragment/Activity is about to be displayed, I recreate my stack again.

[Deleted User] <[Deleted User]> #76 Feb 9

Had the same issue with the view pager and a fragment with a mapview. Android 7.0 on Samsung S8.
Thanks everyone for investigations. I've ended up doing the following:

In onStop of the fragment:
 - call removeAllViews() on instance of the MapView
 - remove the MapView from the layout
In onStart of the fragment:
 - reconstruct the MapView
 - add to the layout
 - setup all the callbacks as usual

I've tried putting this in onDestroyView, it didn't work.
I didn't like using removeAllViews by itself as I don't know what is being removed. The map was left blank so couldn't reconstruct just from that state.
Just removing the MapView without removing it's views first didn't work. I think it's a timing issue. I guess GC is not removing the map view from memory straightaway.

Hopefully Google will address this issue.

[Deleted User] <[Deleted User]> #77 Feb 9

Simply adding android:hardwareAccelerated="false" to the affected <activity> in the AndroidManifest has fixed the problems for us. No other code changes required. Not nice but better than devices.

---

li...@addcn.com <li...@addcn.com> #78 Feb 1

So,How can I solve it?

---

po...@gmail.com <po...@gmail.com> #79 Feb 2

@#76: Could you please elaborate on " reconstruct the MapView" or post some example code? What is meant by "reconstruct". Do I also have to call getMapAsync() again. And how to add th again? Would be so nice to get this fixed / create a working workaround! Thanks!

---

pa...@gmail.com <pa...@gmail.com> #80

Mar 29

I saw it on Galaxy which update to android 7 from lower.

backtrace:
```
  #00  pc 0000000000004ad30  /system/lib/libc.so (tgkill+12)
  #01  pc 00000000000484c3  /system/lib/libc.so (pthread_kill+34)
  #02  pc 000000000001dd99  /system/lib/libc.so (raise+10)
  #03  pc 0000000000019521  /system/lib/libc.so (__libc_android_abort+34)
  #04  pc 0000000000017160  /system/lib/libc.so (abort+4)
  #05  pc 000000000000c6bb  /system/lib/libcutils.so (__android_log_assert+114)
  #06  pc 00000000000274bf  /system/lib/libhwui.so
  #07  pc 0000000000024c97  /system/lib/libhwui.so
  #08  pc 0000000000026643  /system/lib/libhwui.so
  #09  pc 0000000000029b01  /system/lib/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+80)
  #10  pc 000000000000e329  /system/lib/libutils.so (_ZN7android6Thread11_threadLoopEPv+144)
  #11  pc 000000000006a56d  /system/lib/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+80)
  #12  pc 0000000000047f93  /system/lib/libc.so (_ZL15__pthread_startPv+22)
  #13  pc 000000000001a161  /system/lib/libc.so (__start_thread+6)
```

Device list:

Galaxy J7 Prime (on7xelte)	242	9.2%
Galaxy J7(2016) (j7xelte)	238	9.0%
Galaxy Note5 (noblelte)	228	8.6%
Galaxy S6 (zeroflte)	227	8.6%
Galaxy J7 Pro (j7y17lte)	156	5.9%
Galaxy S6 Edge (zerolte)	154	5.8%
Galaxy A5(2016) (a5xelte)	142	5.4%
Galaxy J7 Neo (j7velte)	130	4.9%
Galaxy A5(2017) (a5y17lte)	129	4.9%
Galaxy J5 Prime (on5xelte)	99	3.8%
Galaxy A3(2016) (a3xelte)	66	2.5%
Galaxy S6 Edge+ (zenlte)	61	2.3%
Galaxy J5(2017) (j5y17lte)	60	2.3%
Galaxy Tab A 10.1 (gtaxlwifi)	47	1.8%
Galaxy J3(2017) (j3y17lte)	39	1.5%
Galaxy A7(2016) (a7xelte)	38	1.4%
Galaxy A3(2017) (a3y17lte)	34	1.3%
Galaxy A7(2017) (a7y17lte)	27	1.0%
Galaxy J7 Max (j7maxlte)	23	0.9%
Galaxy S6 (zeroflteIgt)	18	0.7%

wb...@gmail.com <wb...@gmail.com> #81 Apr 19

This is a sample application to reproduce the native crash: https://github.com/salivaoverflow/TextureViewOffViewPort. In order to reproduce it you need to run it on a 7.0 device. The mandat this crash happen is when a TRIM_MEMORY_COMPLETE signal happens. Currently it is 100% reproducible on a Nexus 6 running 7.0.0_r14 (you can get the aosp source code and flash it to a another device Galaxy S6 running 7.0. Maybe it is also reproducible on other devices I haven' tested yet. In order to reproduce it you should manually generate some critical memory situation TRIM_MEMORY_COMPLETE will be triggered by ActivityManagerService. I used to ways to do this: (1) through command line send "am send-trim-memory <pid> COMPLETE". (2) Use https://play.google.com/store/apps/details?id=com.tspoon.androidtoolbelt&hl=en_GB.

The problem is that libhwui has some problem handling a TextureView that is created, attached to view tree but never goes to screen (this is common when you are using ViewPager). The sa create such a situation under different scenarios. You can recreate the crash in the sample app via:

(1) Tap on "Inside ViewPager", scroll the first page a little bit but never swipe to the second pager. Press home to put app into background. Create some low memory situation like "am send-t COMPLETE" then the crash will happen.
(2) Tap on "Changing TextView with TextureView", tap on "Hello World", press home and send trim memory signal like above.
(3) Tap on "Relative Layout with Changing TextView", tap on "TextureView below me", background the app and send a trim memory signal.

This is a trace down of how TRIM_MEMORY_COMPLETE causes the crash:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/services/core/java/com/android/server/am/ActivityManagerService.java#20500 Android uses an OOM score mechanism to compute application and use it as a metric to decide which applications to kill. In some memory constraint situations the algorithm will decide that a particular app needs to do a full memory trim, oth http://androidxref.com/7.0.0_r1/xref/frameworks/base/services/core/java/com/android/server/am/ActivityManagerService.java#20762.
The scheduleTrimMemory call is a binder call. The caller ActivityManagerService runs in process system_server and will invoke a binder method inside the application process:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/app/ActivityThread.java#1656. Then:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/app/ActivityThread.java#4968. Then:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/WindowManagerGlobal.java#505. Then:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/ThreadedRenderer.java#278.
The ThreadedRenderer is the class to do hardware rendering. It is just a wrapper of it's native peer: http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/jni/android_view_ThreadedRe nTrimMemory call in ThreadedRenderer.java will trigger the native peer call: http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/jni/android_view_ThreadedRenderer.cpp#591. Rend some functions with macros and the call will trigger the function in CanvasContext.cpp: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/CanvasContext.cpp# level is TRIM_MEMORY_COMPLETE, this line will be called: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/CanvasContext.cpp#676. Then:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/EglManager.cpp#250. Then: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderstate/Re Then: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/GpuMemoryTracker.cpp#76. When it checks the remaining gpu object number it is greater than 0, then the crashing li http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/GpuMemoryTracker.cpp#82.

The code path above is close ended I don't think there is another possible trigger of the crash. The only possible trigger is through TRIM_MEMORY_COMPLETE.

Then the key is to understand why the remaining gObjectSize is greater than 0 when trimming memory. There are three kinds of GPU object with number counting enabled in libhwui.so: Layer Texture. In this scenario a TextureView is leaked and corresponds to a Layer and Texture in libhwui. For TextureView:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/TextureView.java#375.
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/ThreadedRenderer.java#813.

http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/jni/android_view_ThreadedRenderer.cpp#537.
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/RenderProxy.cpp#288.
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/CanvasContext.cpp#690.
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/LayerRenderer.cpp#276.

Why only happening to offscreen TextureView? There is a check here in View.java: http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/View.java#16804. Basically
TextureView itself or it's parent is totally out of viewport. The quickReject method will call a native method in RecordingCanvas.cpp (on 7.0.0 the backing canvas object for hardware ui is Rec
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RecordingCanvas.cpp#226. Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/CanvasState.cpp#292.
there is always a subregion of visible screen. If the input region is totally out of viewport it will return true. Going back to View.java. If it returns true then the draw method will just return inste
being called: http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/View.java#16968. For a TextureView that has requested a hardware layer, if this line is never cal
the abort later.
The explanation:
The render thread keeps a tasks queue and keeps polling a task from it and run it: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/RenderThread.cpp#287. Or
DrawFrameTask which runs at probably every Vsync: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/DrawFrameTask.cpp#85. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/DrawFrameTask.cpp#93. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/DrawFrameTask.cpp#125. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/CanvasContext.cpp#201. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderthread/CanvasContext.cpp#227. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#230. Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#424. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#520.

This operation basically goes through the view tree display list. Suppose eventually the flow traverses the tree and goes to the RenderNode corresponds to a TextureView: then this will be cal
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#422. Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#470.
This line basically copies the staging display list (the display list maintained by UI thread) to display list (the display list maintained by render thread for rendering). Before doing the display li
cleanup the old render thread display list: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#502. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#508. Then the list of strong pointers will be released:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/DisplayList.h#201. When sp<X> is released where X is a smart pointer object, the strong ref count of X will decrease. In this
the hardware layer (DeferredLayerUpdater) of a previously created TextureView: http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/jni/android_view_ThreadedRenderer.cpp#541. If
DeferredLayerUpdater drops to zero then the destructor of it will be called: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/DeferredLayerUpdater.cpp#45. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/renderstate/RenderState.cpp#228. If the wrapped Layer object has zero ref count after this then the destructor of Layer will
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/Layer.cpp#54. Since Layer is a tracked GPU object (extends from GpuMemoryTracker) so this will be called when Layer is d
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/GpuMemoryTracker.h#62. Then this http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/GpuMemoryTracker.
the hardware layer to be recorded as released so when TRIM_MEMORY_COMPLETE happens the abort won't happen.

Here is an explanation why the DeferredLayerUpdater is here: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/DisplayList.h#201. After TextureView gets a hardware layer th
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/TextureView.java#331. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/DisplayListCanvas.java#225. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/jni/android_view_DisplayListCanvas.cpp#198. Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/Recordinc
Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/DisplayList.h#170.

The above explains in normal cases when the abort doesn't happen. When the app is backgrounded a cycle like this will happen and then the Layer will be released and crash won't happen. H
TextureView or it's parent is out of viewport all the time then the out of viewport view's render node won't be in this list: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/Ren
the bottom half of the operations we talked above won't happen. The reason it is not in the list is because, like we talked above,
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/View.java#16968 is never being called.

If that line is not being called, then this won't http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/DisplayListCanvas.java#210. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/jni/android_view_DisplayListCanvas.cpp#187. Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/Recordinc
Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/DisplayList.cpp#91

So the above explains why a TextureView out of viewport causes the abort.

A work around, we can fix the crash by listening to TRIM_MEMORY_COMPLETE in the app, when it happens (the app is already backgrounded), remove TextureViews in the app from view tree
off screen TextureViews being created (like don't use ViewPager).

The reason why removing TextureView from view tree can prevent the crash. When a TextureView is remove from parent this line will be called:
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/TextureView.java#222. Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/
Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/View.java#15269. Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view
Then http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/java/android/view/RenderNode.java#219. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/core/jni/android_view_RenderNode.cpp#142. http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#87.
address is 0 this line will be called: http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#91. Then
http://androidxref.com/7.0.0_r1/xref/frameworks/base/libs/hwui/RenderNode.cpp#508. According to what we have analyzed above removing display list will cause the hardware layer to be

po...@gmail.com <po...@gmail.com> #82 May 14

@#81: Thanks for your comprehensive analysis!

I have a question regarding your workaround to listen to TRIM_MEMORY_COMPLETE and remove TextureViews from the view tree. How should I react when the app is resumed after I have re
from the view root? Do I have to manually restore the view root and if so, how do I do so? If I understand correcty, the app isn't necessarily killed after TRIM_MEMORY_COMPLETE, so it may b
to restore it. Another question would be how to find out which view is the one causing the problem. I do not use TextureViews in my app directly, so how am I able to find out which view may

ib...@gmail.com <ib...@gmail.com> #83 May 15

I also found this problem.It affected many users.I found it always happened low memory through logs.

ki...@gmail.com <ki...@gmail.com> #84 May 23

Hi,
It's been almost two years. My app got affected recently.
All of them occur on Android 8 devices.
The Android system today is too buggy.
Maybe you should stop developing new features.
Take effort to resolve all these problems on Android.

acktrace:
```
 #00  pc 0000000000069d08  /system/lib64/libc.so (tgkill+8)
 #01  pc 000000000001de10  /system/lib64/libc.so (abort+88)
 #02  pc 0000000000007eec  /system/lib64/liblog.so (__android_log_assert+304)
 #03  pc 000000000004b8b8  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread14OpenGLPipeline11swapBuffersERKNS1_5FrameEbRK6SkRectPNS0_9FrameInfoEPb+208
 #04  pc 00000000000492e4  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread13CanvasContext4drawEv+228)
 #05  pc 000000000004c8d4  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread13DrawFrameTask3runEv+184)
 #06  pc 0000000000053090  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+348)
 #07  pc 0000000000011674  /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+280)
 #08  pc 00000000000b7894  /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+136)
 #09  pc 00000000000666a0  /system/lib64/libc.so (_ZL15__pthread_startPv+36)
 #10  pc 000000000001f1a4  /system/lib64/libc.so (__start_thread+68)
```

[Deleted User] <[Deleted User]> #85 May 3(

@#82 The approach we eventually took is not removing all TextureViews from view tree. We just get ride of the bottom tab plus view pager pattern and switched to plain Fragment replacing. instance we receive dropped to about 10-15% compared to previous release. I don't have a scalable solution of the removing TextureView from view tree pattern. But it indeed can prevent the happening. Another way to look at this crash is that it indicates memory leaks in your app since TRIM_MEMORY_COMPLETE will only be send when the device is under high memory pressure TRIM_MEMORY_COMPLETE is the only possible trigger of the crash. You can consider address this crash by gradually fixing memory leaks in your app. I don't know a way to find out the leak application layer. What I did was putting instrumentation code inside the aosp to track the GPU objects.

ad...@meltingsource.com <ad...@meltingsource.com> #86 May 3(

#85 and #81. I'd like to thank you all for your analysis and comments. Also #85.
But what exactly do you guys mean by "removing TextureView from the view tree"?
I mean, #68 already does [root.removeView(textureView)] in his code to demonstrate the crash (it still happens).
I've tried removing all my TextureViews either on onStop or on TRIM_MEMORY_COMPLETE, to no avail.

Let's assume I need to use a ViewPager with TextureViews. I can also easily rebuild everything from scratch if needed be. How can I safely dispose of the ViewPager/TextureViews to avoid th

This crash accounts for 50% of my app's crashes. I guess it will go away when 7.0 goes away.

ga...@gmail.com <ga...@gmail.com> #87 May 3(

This crash is one of the reasons why I started focusing more attention on
iOS.
On Wed, 30 May 2018 at 8:39 pm, <buganizer-system@google.com> wrote:

- Show quoted text -

wb...@gmail.com <wb...@gmail.com> #88 Jun 1

#86 Removing TextureView from view tree means:

```
ViewGroup vg = textureView.getParent();
vg.removeView(textureView);
```

One certain thing is that TRIM_MEMORY_COMPLETE is the only possible trigger of the crash. However there are more than one possible scenarios that could result in the "crashable" state of TRIM_MEMORY_COMPLETE is emitted. The scenario that I discovered where there is a TextureView attached to view tree but never goes into screen is one of them. The scenario #68 discove pending animation when removing TextureView from view tree is another scenario. Of course there could be other scenarios to result in the bad state of libhwui but the only possible trigger t crash happen is TRIM_MEMORY_COMPLETE signal. Personally I think the scenario I discovered is more common since if you use ViewPager and TextureView together this scenario is very ea

Whether you can safely get rid of ViewPager depends on your app logic. There is a pattern where BottomNavigationView is used together with ViewPager and the ViewPager is made non-swi ViewPager is not swippable anyways you can choose no to use ViewPager. If the swipe behavior is a necessity then of course you cannot get rid of ViewPager.

This aborting code in libhwui is added in 7.0 so for sure it won't happen before 7.0. For newer versions the code is still there I am not sure about the crash rate on later versions. I analyzed it

Note that the only possible trigger is TRIM_MEMORY_COMPLETE, you can argue that this is low user impact since it happens when the app is backgrounded. And note that this means there i pressure so it's quite possible that there are some bad memory leaks in your app. You can use this tool to assist reproducing this crash: https://play.google.com/store/apps/details?id=com.tspoon.androidtoolbelt&hl=en_US

po...@gmail.com <po...@gmail.com> #89 Jun 1

#88: I am pretty bound to ViewPager, don't think I can get rid of it.

Regarding TRIM_MEMORY_COMPLETE, how do you conclude there may be memory leaks in my app. Isn't this signal just send to all background processes in case the system gets under mer it could just be a memory intensive game which gets loaded in that moment or another app with many leaks.

wb...@gmail.com <wb...@gmail.com> #90 Jun 2

#89 I didn't say for sure there is a memory leak in your app. There is a big memory pressure on your device for sure. If you are observing many crashes then it's quite possible that there is me app. At least I think it is worth looking at. In my case there is indeed a serious memory leak which leaks every instance of a certain Activity.

pu...@gmail.com <pu...@gmail.com> #91 Jun 28

My app use heavily a swipeable ViewPager.

Over a period of 2 months, I have 649 reports impacting 453 users.
100% of devices are various Samsung models all running Android 8.0:

```
  #00  pc 0000000000071854  /system/lib64/libc.so (tgkill+8)

  #01  pc 000000000001e058  /system/lib64/libc.so (abort+88)

  #02  pc 0000000000008248  /system/lib64/liblog.so (__android_log_assert+328)

  #03  pc 0000000000052430  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread10EglManager11damageFrameERKNS1_5FrameERK6SkRect+320)

  #04  pc 000000000004f9dc  /system/lib64/libhwui.so
(_ZN7android10uirenderer12renderthread14OpenGLPipeline4drawERKNS1_5FrameERK6SkRectS8_RKNS0_12FrameBuilder13LightGeometryEPNS0_16LayerUpdateQueueERKNS0_4RectEbRl
derer9LightInfoERKNSt3__16vectorINS_2spINS0_10RenderNodeEEENSM_9allocatorISQ_EEEEPNS0_19FrameInfoVisualizerE+76)

  #05  pc 000000000004d7e0  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread13CanvasContext4drawEv+176)

  #06  pc 00000000000511e8  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread13DrawFrameTask3runEv+184)

  #07  pc 0000000000058494  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+356)

  #08  pc 0000000000011c58  /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+280)

  #09  pc 00000000000fd688  /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+136)

  #10  pc 000000000006de44  /system/lib64/libc.so (_ZL15__pthread_startPv+36)

  #11  pc 000000000001f9a4  /system/lib64/libc.so (__start_thread+68)
```

hu...@gmail.com <hu...@gmail.com> #92 Aug 1(

Fix the memory leaks does not help for this bug

se...@gmail.com <se...@gmail.com> #93 Aug 1(

After the analysis done by #81 (thx again) I've tried to implement a workaround by listening to TRIM_MEMORY_COMPLETE and reacting on it. Removing the ViewPager is not a possibility in n
have done: Listening to TRIM_MEMORY_COMPLETE and kill my app in case the devices Android version is 7.0 (I am only seeing the crash there) and not all of the relevant ViewPager views h
(as according to the analysis the crash shouldn't happen otherwise). I thought it would be better to just kill it and therefore preventing it from "officially crashing" and polluting my stats in And
complete restart without saving the current activity state and so on needs to be done anyway.
This workaround successfully prevented the crash BUT I could not use it anyway because of another problem. TRIM_MEMORY_COMPLETE is not only sent if the app is backgrounded! I repro
myself on real devices and my users had this effect to after I published the app update in Google Play where TRIM_MEMORY_COMPLETE was received by my app WHILE BEEING IN THE FOR
workaround this of course leads to my app bering killed (by myself) although it is still bering used. I don't know if this behavior is a bug or not but it's there and makes it impossible for me to

Any held or updates are very much appreciated!

la...@gmail.com <la...@gmail.com> #94 Aug 2

```
android 8.0 application getting crash
 Build fingerprint: 'google/angler/angler:8.0.0/OPR6.170623.019/4299446:user/release-keys'
    Revision: '0'
    ABI: 'arm'
    pid: 25501, tid: 26158, name: Thread-66  >>> com.tru <<<
    signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
08-21 18:32:53.162 26343-26343/? A/DEBUG: Abort message: 'invalid pthread_t 0xc79aa970 passed to libc'
        r0 00000000  r1 0000662e  r2 00000006  r3 00000008
        r4 0000639d  r5 0000662e  r6 c7455788  r7 0000010c
        r8 0000639d  r9 ad47f138  sl a9a24c74  fp c74557dc
        ip 00000000  sp c7455778  lr eec5d3b7  pc eec8d91c  cpsr 200f0010
08-21 18:32:53.167 26343-26343/? A/DEBUG: backtrace:
        #00 pc 0004a91c  /system/lib/libc.so (tgkill+12)
        #01 pc 0001a3b3  /system/lib/libc.so (abort+54)
        #02 pc 0001e999  /system/lib/libc.so (__libc_fatal+24)
        #03 pc 00047f23  /system/lib/libc.so (_Z23__pthread_internal_findl+82)
        #04 pc 00047c49  /system/lib/libc.so (pthread_detach+4)
```

po...@gmail.com <po...@gmail.com> #95 Aug 2

@#94: That is a different kind of crash! Got nothing to do with the problem discussed in this thread.

[Deleted User] <[Deleted User]> #96 Sep 3(

I have ViewPager with CameraView in first page. If first page is opened and app go to background, all right. But, if second page is opened (so, CameraView is invisible) and app go to backgro
crashed with Signal 6 in RenderThread. So, my hack:

```
@Override
protected void onPause() {
    if(viewPager != null && viewPager.getCurrentItem() == 1) {
        isCurrentPageIsSecond = true;
        viewPager.setCurrentItem(0);
    }
    super.onPause();
}

@Override
protected void onResume() {
    super.onResume();
    if(viewPager != null && isCurrentPageIsSecond) {
        isCurrentPageIsCollection = false;
        viewPager.setCurrentItem(1);
    }
}
```

I tested it with 'adb shell am send-trim-memory org.rxlab.photonotes COMPLETE' and it no crashes.

is...@google.com <is...@google.com>

Status: New

ga...@gmail.com <ga...@gmail.com> #97 Oct 5

Is this finally being looked at? I stopped developing for Android because of this issue. This was the last straw to make me jump to developing for iOS.

al...@gmail.com <al...@gmail.com> #98 Oct 27

Any updates on this issue, have been facing same crash on Samsung Galaxy series with Android 7.0 and 7.1. My app has google maps, inside Viewpager. 1st activity has Viewpager with two
and other with a list using RecyclerView. Upon clicking list item new Activity is opened and it has the option to take pictures and when camera Intent is launched after 1 or 2 secs my app cras

al...@gmail.com <al...@gmail.com> #99 Oct 27

here is a backtrace that I see and play console

Samsung Galaxy S6 Edge (zeroltevzw), Android 7.0
Report 1

*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
pid: 0, tid: 0 >>> <applicationId><<<

backtrace:

```
#00  pc 000000000006b5b4  /system/lib64/libc.so (tgkill+8)
#01  pc 0000000000068a50  /system/lib64/libc.so (pthread_kill+64)
#02  pc 0000000000023f68  /system/lib64/libc.so (raise+24)
#03  pc 000000000001c9ec  /system/lib64/libc.so (abort+52)
#04  pc 00000000000112ec  /system/lib64/libcutils.so (__android_log_assert+232)
#05  pc 0000000000057700  /system/lib64/libhwui.so
#06  pc 00000000000372f4  /system/lib64/libhwui.so
#07  pc 0000000000034ad0  /system/lib64/libhwui.so
#08  pc 0000000000039620  /system/lib64/libhwui.so
#09  pc 000000000003a720  /system/lib64/libhwui.so
#10  pc 000000000003bb5c  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+148)
#11  pc 0000000000012488  /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
#12  pc 00000000000a4bb0  /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
#13  pc 0000000000068258  /system/lib64/libc.so (_ZL15__pthread_startPv+196)
#14  pc 000000000001dc00  /system/lib64/libc.so (__start_thread+16)
```

al...@gmail.com <al...@gmail.com> #100 Dec 3

any updates below is the backtrace from when device is attached to the debugger

```
2018-12-03 11:28:45.592 17280-17280/? A/DEBUG: Build fingerprint: 'samsung/zeroltedv/zerolte:7.0/NRD90M/G925IDVS3FRB1:user/release-keys'
2018-12-03 11:28:45.593 17280-17280/? A/DEBUG: Revision: '10'
2018-12-03 11:28:45.593 17280-17280/? A/DEBUG: ABI: 'arm64'
2018-12-03 11:28:45.593 17280-17280/? A/DEBUG: pid: 8251, tid: 8316, name: RenderThread  >>> <applicationId> <<<
2018-12-03 11:28:45.593 17280-17280/? A/DEBUG: signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG: Abort message: 'Leaked 2 GPU objects!'
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:     x0  0000000000000000 x1  000000000000207c x2  0000000000000006 x3  0000000000000008
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:     x4  0000000000000021 x5  0000000000000000 x6  000000738bbdc000 x7  0000000000000000
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:     x8  0000000000000083 x9  fffffffffffffdf x10 0000000000000000 x11 0000000000000001
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:     x12 ffffffffffffffff x13 0000000000000000 x14 0000000000000000 x15 0034f521fa871c1b
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:     x16 0000007387c89ee0 x17 0000007387c335ac x18 0000000000000000 x19 00000073620824f8
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:     x20 0000000000000006 x21 0000007362082450 x22 000000000000000b x23 0000007389d3c008
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:     x24 00000073365c7760 x25 00000073873d5060 x26 000000733a24f920 x27 00000073873d5078
```

```
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:    x28  7fffffffffffffff  x29  0000007362081af0  x30  0000007387c30a54
2018-12-03 11:28:45.604 17280-17280/? A/DEBUG:    sp   0000007362081ad0  pc   0000007387c335b4  pstate 0000000060000000
2018-12-03 11:28:45.639 17280-17280/? A/DEBUG: backtrace:
2018-12-03 11:28:45.641 17280-17280/? A/DEBUG:    #00 pc 000000000006b5b4  /system/lib64/libc.so (tgkill+8)
2018-12-03 11:28:45.641 17280-17280/? A/DEBUG:    #01 pc 0000000000068a50  /system/lib64/libc.so (pthread_kill+64)
2018-12-03 11:28:45.641 17280-17280/? A/DEBUG:    #02 pc 000000000023f68  /system/lib64/libc.so (raise+24)
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #03 pc 000000000001c9ec  /system/lib64/libc.so (abort+52)
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #04 pc 00000000000112ec  /system/lib64/libcutils.so (__android_log_assert+232)
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #05 pc 0000000000057700  /system/lib64/libhwui.so
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #06 pc 00000000000372f4  /system/lib64/libhwui.so
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #07 pc 0000000000034ad0  /system/lib64/libhwui.so
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #08 pc 0000000000039620  /system/lib64/libhwui.so
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #09 pc 000000000003a720  /system/lib64/libhwui.so
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #10 pc 000000000003bb5c  /system/lib64/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+148)
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #11 pc 0000000000012488  /system/lib64/libutils.so (_ZN7android6Thread11_threadLoopEPv+272)
2018-12-03 11:28:45.643 17280-17280/? A/DEBUG:    #12 pc 00000000000a4bb0  /system/lib64/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+116)
2018-12-03 11:28:45.644 17280-17280/? A/DEBUG:    #13 pc 0000000000068258  /system/lib64/libc.so (_ZL15__pthread_startPv+196)
2018-12-03 11:28:45.644 17280-17280/? A/DEBUG:    #14 pc 000000000001dc00  /system/lib64/libc.so (__start_thread+16)
```

mk...@gmail.com <mk...@gmail.com> #101 Dec 1(

Same happen in Android 7.0 for many different devices.

```
 #00  pc 000000000004a230  /system/lib/libc.so (tgkill+12)
 #01  pc 00000000000479c3  /system/lib/libc.so (pthread_kill+34)
 #02  pc 000000000001d9c5  /system/lib/libc.so (raise+10)
 #03  pc 0000000000019511  /system/lib/libc.so (__libc_android_abort+34)
 #04  pc 0000000000017150  /system/lib/libc.so (abort+4)
 #05  pc 000000000000c5c7  /system/lib/libcutils.so (__android_log_assert+114)
 #06  pc 000000000003cf15  /system/lib/libhwui.so
 #07  pc 0000000000027183  /system/lib/libhwui.so
 #08  pc 0000000000025761  /system/lib/libhwui.so
 #09  pc 00000000000281d1  /system/lib/libhwui.so
 #10  pc 0000000000028a73  /system/lib/libhwui.so
 #11  pc 0000000000029aa1  /system/lib/libhwui.so (_ZN7android10uirenderer12renderthread12RenderThread10threadLoopEv+80)
 #12  pc 000000000000e401  /system/lib/libutils.so (_ZN7android6Thread11_threadLoopEPv+144)
 #13  pc 000000000006afb5  /system/lib/libandroid_runtime.so (_ZN7android14AndroidRuntime15javaThreadShellEPv+80)
 #14  pc 0000000000047493  /system/lib/libc.so (_ZL15__pthread_startPv+22)
 #15  pc 0000000000019f6d  /system/lib/libc.so (__start_thread+6)
```

al...@gmail.com <al...@gmail.com> #102 Dec 1

android-bugreport@google.com at least provide some official response that there is an issue and whether or not its being looked up.

ho...@gmail.com <ho...@gmail.com> #103 Dec 1

How to resolve this problem ? tks

```
11-27 17:31:10.942438 30334 30334 F DEBUG   : ABI: 'arm64'
11-27 17:31:10.942474 30334 30334 F DEBUG   : pid: 2692, tid: 3034, name: RenderThread  >>> com.android.systemui <<<
11-27 17:31:10.942508 30334 30334 F DEBUG   : signal 6 (SIGABRT), code -6 (SI_TKILL), fault addr --------
11-27 17:31:10.942559 30334 30334 F DEBUG   : Abort message: 'sp<> assignment detected data race'
11-27 17:31:10.942619 30334 30334 F DEBUG   :     x0  0000000000000000  x1  0000000000000bda  x2  0000000000000006  x3  0000000000000008
11-27 17:31:10.942656 30334 30334 F DEBUG   :     x4  fefeff741ebf1d7f  x5  fefeff741ebf1d7f  x6  fefeff741ebf1d7f  x7  7f7f7f7f7fff7fff
11-27 17:31:10.942691 30334 30334 F DEBUG   :     x8  0000000000000083  x9  000000751fc3a588  x10 ffffffff87ffffbdf  x11 0000000000000001
11-27 17:31:10.942725 30334 30334 F DEBUG   :     x12 00000074822fae30  x13 ffffffffffffffff  x14 fffffffffff000000  x15 ffffffffffffffff
11-27 17:31:10.942791 30334 30334 F DEBUG   :     x16 000000751fc712b0  x17 000000751fbaf3d8  x18 00000074822fa61a  x19 0000000000000a84
11-27 17:31:10.942833 30334 30334 F DEBUG   :     x20 0000000000000bda  x21 0000000000000083  x22 00000074808b3ca0  x23 00000074808b3cb8
11-27 17:31:10.942873 30334 30334 F DEBUG   :     x24 000000749e8f7310  x25 00000074809b9760  x26 000000747b5603e0  x27 431bde82d7b634db
11-27 17:31:10.942912 30334 30334 F DEBUG   :     x28 0000000000000000  x29 00000074822fad50
11-27 17:31:10.942946 30334 30334 F DEBUG   :     sp  00000074822fad10  lr  000000751fba3b90  pc  000000751fba3bbc
11-27 17:31:10.998488 30334 30334 F DEBUG   :
11-27 17:31:10.998488 30334 30334 F DEBUG   : backtrace:
11-27 17:31:10.998566 30334 30334 F DEBUG   :     #00 pc 0000000000021bbc  /system/lib64/libc.so (abort+124)
11-27 17:31:10.998601 30334 30334 F DEBUG   :     #01 pc 00000000000080f8  /system/lib64/liblog.so (__android_log_assert+296)
11-27 17:31:10.998635 30334 30334 F DEBUG   :     #02 pc 000000000000f26c  /system/lib64/libutils.so (android::sp_report_race()+28)
11-27 17:31:10.998676 30334 30334 F DEBUG   :     #03 pc 0000000001100d0  /system/lib64/libandroid_runtime.so (android::RootRenderNode::detachAnimators()+320)
11-27 17:31:10.998710 30334 30334 F DEBUG   :     #04 pc 000000000010e1a0  /system/lib64/libandroid_runtime.so (android::AnimationContextBridge::destroy()+24)
11-27 17:31:10.998746 30334 30334 F DEBUG   :     #05 pc 00000000005d182c  /system/lib64/libhwui.so (android::uirenderer::renderthread::CanvasContext::destroy()+124)
11-27 17:31:10.998780 30334 30334 F DEBUG   :     #06 pc 00000000005d4c60  /system/lib64/libhwui.so (std::__1::packaged_task<void ()>::operator()()+88)
11-27 17:31:10.998816 30334 30334 F DEBUG   :     #07 pc 0000000000587918  /system/lib64/libhwui.so (android::uirenderer::WorkQueue::process()+168)
11-27 17:31:10.998850 30334 30334 F DEBUG   :     #08 pc 00000000001fd9b4  /system/lib64/libhwui.so (android::uirenderer::renderthread::RenderThread::threadLoop()+244)
11-27 17:31:10.998884 30334 30334 F DEBUG   :     #09 pc 000000000000fb80  /system/lib64/libutils.so (android::Thread::_threadLoop(void*)+280)
11-27 17:31:10.998923 30334 30334 F DEBUG   :     #10 pc 000000000008495c  /system/lib64/libc.so (__pthread_start(void*)+36)
11-27 17:31:10.998961 30334 30334 F DEBUG   :     #11 pc 00000000000234bc  /system/lib64/libc.so (__start_thread+68)
```

al...@gmail.com <al...@gmail.com> #104 Jan 1

any solution??

an...@gmail.com <an...@gmail.com> #105 Jan 1

I was getting this issue in my Samsung Galaxy J7 2016 (Nougat). But upgrading to Oreo fixed this for me.
You can see [here] (https://stackoverflow.com/questions/53558154/sigabrt-crash-in-samsung-devices) that pattern is common with samsung device and API 7.

---

mi...@gmail.com <mi...@gmail.com> #106 Apr 1

@#81 Thanks for your detailed analysis. Could you provide the fix change in Android O? I'm eager to learn how to fix it in system layer. Thank you very much!

---

hh...@gmail.com <hh...@gmail.com> #107 Jun

@#81 ,Thank for your details , and the demo crash on emulator 100% 。

---

cc...@google.com <cc...@google.com> #108 Jul

Status: Won't Fix (Obsolete)

Thank you for your feedback. Unfortunately, this issue is not reproducible with our version of the AOSP source code for Nexus/Pixel devices. Please file a bug with the respective OEM, as the customized version of the AOSP source code. This bug will be marked as Obsolete.

---

ji...@gmail.com <ji...@gmail.com> #109 Nov

Comment has been deleted.

Message last modified on Nov 17, 2021 02:26PM