Release version of a native library from a dependent project gets packaged even for a debug build variant

DESCRIPTION be...@gmail.com created issue #1

Originally filed: https://github.com/android-ndk/ndk/issues/197#issuecomment-246448343

Setup. 2 projects `app` project and `libModule`. This bug has 2 parts which I think are related so I'm putting both parts here.

Part 1: Can't debug library project, setting breakpoints do not work. And when a segfault happens ndk breaks in "unknown function` and nothing useful can be done.

A workaround solution is in the main project do this:

```
    // default config
    externalNativeBuild {
      cmake {
        abiFilters'armeabi-v7a'
        arguments "-DANDROID_PLATFORM=android-${platformVersion}",
            '-DANDROID_TOOLCHAIN=clang', '-DANDROID_STL=gnustl_shared'
      }
    }
  // in android {}
  externalNativeBuild {
    cmake {
      path "../libModule/src/main/cpp/CMakeLists.txt"
    }
  }
```

and I can debug just fine. So far it's fine. This has the disadvantage of compiling everything twice.

part 2: Library project always compiles in the release folder `.externalNativeBuild` regardless what has been specified. Interesting though if I do "Build->build APK" in android studio RC2 it bui

Some discussion [here](https://github.com/googlesamples/android-ndk/issues/238#issuecomment-246367381)

Some version specifiers

```
Android Studio 2.2 RC 2
Build #AI-145.3253452, built on September 7, 2016
JRE: 1.8.0_76-release-b03 x86_64
JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

classpath 'com.android.tools.build:gradle:2.2.0-rc2'

compileSdkVersion 24
buildToolsVersion "24.0.2"

      cmake {
        abiFilters'armeabi-v7a'
        arguments "-DANDROID_PLATFORM=android-19",
            '-DANDROID_TOOLCHAIN=clang', '-DANDROID_STL=gnustl_shared'
      }
```

Studio Build: RC2
Version of Gradle Plugin: 2.2.0-rc2
Version of Gradle: 2.14.1
Version of Java: 8
OS: Mac

Steps to Reproduce: make a project with app & module. Module is C++ with cmake. Put breakpoint in C++ code or have C++ code purposely crash. Try debugging. You can't debug.

✓ Links (7)

Links (7)

“Originally filed: https://github.com/android-ndk/ndk/issues/197#issuecomment-246448… ”
“Some discussion [here](https://github.com/googlesamples/android-ndk/issues/238#issuecomment-246367381)”

“ https://github.com/googlesamples/android-vulkan-tutorials/tree/master/tutorial… ”

“Specifically, https://github.com/android-ndk/ndk/issues/197#issuecomment-468988738 solves my current issue.”

“ … back in Android Studio Chipmunk 2021.2.1 with AGP 7.2.0 and Gradle 7.4.2. I've attached an updated version of the hello-jni project that demonstrates the problem; only fix appears to be manuall

See all related links

COMMENTS

ks...@google.com <ks...@google.com>

Assigned to la...@google.com.

la...@google.com <la...@google.com> #2

Hi,

thanks for the report. It sounds like the debugger is having problems finding the symbol files from the other projects.

To help us understand the issue, it would be best if you could create a sample test project demonstrating the problem. Could you create a zip with a small test project?

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

I took the sample code hello-jni and moved it to a library project. Attached hello-jni. As you can see you can't debug into the library. When I set breakpoints they are completely ignored.

hello-jni 2.zip
896 KB Download

be...@gmail.com <be...@gmail.com> #4

Also you can see it builds only release of the mylibrary project.

la...@google.com <la...@google.com> #5

Thanks a lot. I'll look into this.

la...@google.com <la...@google.com> #6

Reassigned to jo...@google.com.

OK, I agree with your assessment. The problem is that we are packaging the release version of the library into the apk. This is more of a build system than a debugger issue. Reassigning the

Jomo: do you have an idea why is this happening?

gf...@google.com <gf...@google.com> #8

The following solution works fine:
1) in app's build.gradle, change dependency to
compile project(path: ':layerlib', configuration: 'debug')
so debug lib build will be used
2) on lib's build.gradle, enable debug version lib to be built and copied (published) to apk
android.publishNonDefault true

added to vulkan validation's layer build:
https://github.com/googlesamples/android-vulkan-tutorials/tree/master/tutorial03_traceable_layers

be...@gmail.com <be...@gmail.com> #9

With the workaround it's impossible to set CMAKE_BUILD_TYPE variable to Debug/Release. It's always Release yet builds with debug flags. Also you cannot build Release unless you modify t

be...@gmail.com <be...@gmail.com> #10

I see, CMAKE_BUILD_TYPE is changeable I had an error in my code. The second part still stands about switching debug/release.

jo...@google.com <jo...@google.com> #11

I believe we're hitting the limitation that gradle doesn't communicate product flavor from lib to app. We have the same problem in java but it is aggravated by the larger difference between de

xa...@google.com <xa...@google.com> #14

Marked as fixed, reassigned to xa...@google.com.

Gradle Android plugin 3.0 now do proper variant aware dependency resolution.

je...@gmail.com <je...@gmail.com> #15

Really? Android Studio 3.5.0 and Android Gradle Plugin 3.5.0 seem to have an issue with CMake in this regard -- but not due to building/incorporating the release .so's. Rather, it seems to be

Specifically, https://github.com/android-ndk/ndk/issues/197#issuecomment-468988738 solves my current issue.

It took me hours of frustration to find this helpful comment. I'm going to open a new bug on Android Studio 3.5.0 accordingly.

Je...@envrmnt.com <Je...@envrmnt.com> #16

This issue appears to be back in Android Studio Chipmunk 2021.2.1 with AGP 7.2.0 and Gradle 7.4.2. I've attached an updated version of the hello-jni project that demonstrates the problem;

hello-jni_2022.zip
8.3 MB Download

---

em...@google.com <em...@google.com> #17

Status: Assigned (reopened)

Thanks for reporting this issue. I am able to reproduce the issue easily.

---

solodkyy@ Queries for dependent modules returns incorrect results in Chipmunk. Specifically, using the sample project in comment#16, the query for modules that `app` depends on returns e

This causes the following (both P0) problems:

1. "Auto" debugger decides to launch Java-only instead of Java+Native
   - problem is here
   - Screenshot: https://screenshot.googleplex.com/BBKzGa7pT8WMYXw
2. Native debugger cannot find the debugging symbols of `mylibrary`.
   - problem is here
   - Screenshot: https://screenshot.googleplex.com/5rvcPaDAW9mRbUr

// Note: The links above are Google-internal.

Message last modified on May 24, 2022 03:18AM

---

so...@google.com <so...@google.com>

Reassigned to sm...@google.com.

---

sm...@google.com <sm...@google.com>

Marked as fixed.

---

em...@google.com <em...@google.com> #18

This issue (about missing debug symbols for library modules) has been fixed and the fix will be included in Chipmunk Patch 2.

---

ve...@google.com <ve...@google.com> #19

This issue is still reproducible in Chipmunk Patch 2.

Scenario-1:
1. Release Version abi's are displaying in debug apk too.

Scenario-2:
1. While trying to debug the sample project with breakpoints in both library and app modules , App is crashed with java.lang.UnsatisfiedLinkError

stack trace:

```
2022-07-27 14:28:20.537 7370-7370/com.example.hellojni E/AndroidRuntime: FATAL EXCEPTION: main
   Process: com.example.hellojni, PID: 7370
   java.lang.UnsatisfiedLinkError: dlopen failed: library "libhello-jni.so" not found
      at java.lang.Runtime.loadLibrary0(Runtime.java:1077)
      at java.lang.Runtime.loadLibrary0(Runtime.java:998)
      at java.lang.System.loadLibrary(System.java:1656)
      at com.example.hellojni.HelloJni.<clinit>(HelloJni.java:59)
      at java.lang.Class.newInstance(Native Method)
      at android.app.AppComponentFactory.instantiateActivity(AppComponentFactory.java:95)
      at androidx.core.app.CoreComponentFactory.instantiateActivity(CoreComponentFactory.java:45)
      at android.app.Instrumentation.newActivity(Instrumentation.java:1285)
      at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:3578)
      at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:3842)
      at android.app.servertransaction.LaunchActivityItem.execute(LaunchActivityItem.java:103)
      at android.app.servertransaction.TransactionExecutor.executeCallbacks(TransactionExecutor.java:135)
      at android.app.servertransaction.TransactionExecutor.execute(TransactionExecutor.java:95)
      at android.app.ActivityThread$H.handleMessage(ActivityThread.java:2252)
      at android.os.Handler.dispatchMessage(Handler.java:106)
      at android.os.Looper.loopOnce(Looper.java:201)
      at android.os.Looper.loop(Looper.java:288)
```

at android.app.ActivityThread.main(ActivityThread.java:7842)
at java.lang.reflect.Method.invoke(Native Method)
at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:548)
at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:1003)

---

em...@google.com <em...@google.com> #20

If the issue reproduces in Chipmunk Patch 2, we need more detailed instructions on how to reproduce it. This bug diverged from its original title, so I suggest filing a separate bug, and includ

Based on my efforts to reproduce using Chipmunk Patch 2, debugging (with app=non-native, lib=native) works, and no `UnsatisfiedLinkError`s are observed.

---

tu...@gmail.com <tu...@gmail.com> #21

Originally filed: https://github.com/android-ndk/ndk/issues/197#issuecomment-246448343

---

dy...@gmail.com <dy...@gmail.com> #22

Comment has been deleted.

Message last modified on Oct 22, 2022 02:52AM

---

dy...@gmail.com <dy...@gmail.com> #23

Comment has been deleted.

Message last modified on Oct 22, 2022 02:52AM

deleted
0 B

---

cl...@gmail.com <cl...@gmail.com> #24

Comment has been deleted.

Message last modified on Oct 22, 2022 02:51AM

---

ka...@gmail.com <ka...@gmail.com> #25

Android 13

---

ka...@gmail.com <ka...@gmail.com> #26

Funtouch os13

---

cl...@gmail.com <cl...@gmail.com> #27

Comment has been deleted.

Message last modified on Oct 22, 2022 02:51AM