Some classes failing code-shrinking/shaking

WAI Bug P2 + Add Hotlist

STATUS UPDATE No update yet. Edit

DESCRIPTION pa...@willowtreeapps.com created issue #1 May 13, 2020 05:00AM

AGP version: 3.6.3
R8 version: default
minSdk: 23
targetSdk: 29

I have a class which is kept based on a custom annotation that it has using:
`-keepclassmembers @AnnotationClass class * { <fields>; }`

After building the apk and inspecting the dex files, the offending class is missing its fields and all of its method's bytecode is similar to:

```
.method public equals(Ljava/lang/Object;)Z
    .registers 2
    const-string p0, "[R8]"
    const-string p1, "Shaking error: Missing method in [REDACTED]: boolean equals(java.lang.Object)"
    invoke-static {p0, p1}, Landroid/util/Log;->e(Ljava/lang/String;Ljava/lang/String;)I
    new-instance p0, Ljava/lang/RuntimeException;
    invoke-direct {p0, p1}, Ljava/lang/RuntimeException;-><init>(Ljava/lang/String;)V
    throw p0
.end method
```

If I keep the class's members explicitly with:
`-keepclassmembers class MyClass { <fields>; }`
or even:
`-keepclassmembers class MyClass { *; }`
Then the fields are still missing and the above shaking error is still present in the byte code

Fully keeping the class, however, does seem to work:
`-keep class MyClass`

I should also note that this is a multimodule project, and this class implements an interface, however the other class that implements the same interface does not appear to have this issue.

I am unable to recreate this issue in a sample project, and as far as I can tell, this class is no different than the hundreds of others we have that follow this same pattern and don't exhibit this issue.

This is a production app and we have restrictions around sharing the code, however I can share an obfuscated apk, though I would much prefer it to be in a private channel if possible.

I also see no behavior change with or without `-addconfigurationdebugging`, I've seen that flag cause issues before so I figure I'd call it out.

COMMENTS All comments ↓ Oldest first

ze...@google.com <ze...@google.com> #2 May 13, 2020 02:43PM

Reassigned to ze...@google.com.

Thanks for the report.

The rule "-keepclassmembers" only applies if the class is retained in some way, whereas "-keep" will keep the class and members regardless of the content of the program.

I suspect the cause is because nothing in the application is instantiating the class. If the class is allocated by reflection or JNI you should keep the constructor(s) that are used in those places, which will cause the class to be seen as instantiated:

-keep class MyClass { <init>(...); }

(Using -keep class MyClass, will in non-full mode implicitly keep the default constructor of the class.)

And then also add rules for the members that are accessed via reflection or JNI (such as the fields as you indicated).

pa...@willowtreeapps.com <pa...@willowtreeapps.com> #3 May 14, 2020 01:03AM

Does it even require instantiation to keep the class? I ask because we have many classes that are only instantiated through reflection, however their type is at least explicitly referenced somewhere else, I'm seeing now that this class never has its type explicitly referenced, it's only referenced through the interface it implements. Wonder if that's the issue here

Reporter	pa...@willowtreeapps.com
Type	Bug
Priority	P2
Severity	S2
Status	Won't fix (Intended behavior)
Access	Default access View
Assignee	ze...@google.com
Verifier	--
Collaborators	
CC	pa...@willowtreeapps.com r8...@googlegroups.com sg...@google.com
AOSP ID	--
Blocking Release	--
Release Status	--
Found In	--
Targeted To	--
Verified In	--
In Prod	

Show 1 additional field

sg...@google.com <sg...@google.com> #4 May 14, 2020 05:43PM

As Ian already mentioned: "Using -keep class MyClass, will in non-full mode implicitly keep the default constructor of the class.". The same is the case when tracing the code. In non-full mode a referenced type will keep the class and at least its default constructor. In full mode the default constructor will not be kept implicitly.

And for the `MyClass` you should see that having a reference to the type (e.g. `System.out.println(MyClass.class)` will make the `-keepclassmembers` rules take effect. In general it is good practice to have an explicit `-keep` rule on all classes which are instantiated by reflection.

---

ro...@gmail.com <ro...@gmail.com> #5 May 14, 2020 11:19PM

Alright great! Yea it seems this was just a misunderstanding as to how `-keepclassmembers` worked, I wasn't aware that the class needed to already be kept for that to take effect. Thanks for the prompt response y'all, our issue is resolved and this doesn't appear to be an issue with R8 itself. Hoping the next person that runs into this finds this and it's able to help them.

Thanks!

---

ze...@google.com <ze...@google.com> #6 May 15, 2020 06:22AM

Status: Won't Fix (Intended Behavior)

Thanks for the update and we are happy to hear the issue is resolved. Don't hesitate to file future issues when you encounter them!