IssueTracker Search IssueTracker Sign in

Android Public Tracker > App Development > Android Studio > Editing > C++ Editor 266174935

[Giraffe | 2022.3.1 Canary 2] On opening CPP file, showing infinite loader in editor.

+1 10 Hotlists (7) Mark as Duplicate

Comments (19) Dependencies Duplicates (6) Blocking (0) Resources (6)

Fixed Bug P1 +

STATUS UPDATE No update yet. Edit

DESCRIPTION sb...@google.com created issue #1

Studio Version Numer: Android Studio Giraffe | 2022.3.1 Canary 2
Build #AI-223.7571.182.2231.9506613, built on January 19, 2023
Runtime version: 17.0.5+0-b2043.56-9484017 amd64
VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.
Windows 10 10.0
GC: G1 Young Generation, G1 Old Generation
Memory: 4096M
Cores: 8
Registry:
  external.system.auto.import.disabled=true
  ide.text.editor.with.preview.show.floating.toolbar=false
  ide.instant.shutdown=false

Regression: Yes (Electric Eel | 2022.1.1)
Repro: 3/3
Found in: SANITY
Platform: All

For C++ Projects:
NDK Version: 25.1.8937393
Cmake Version: 3.22.1

Steps to Reproduce:
1. Create a Native C++ project.
2. On successful build, open CPP folder -> any .cpp file in the editor.

Expected Results:
The file should the content in the editor

Observed Results:
Opened CPP file is Showing infinite loader, failing not loading the content in the editor.
In the Studio at bottom, showing "Context: Indexing" for long time.

idea.log
3.8 MB View Download

Untitled_ Jan 20, 2023 1_45 PM.webm
7.9 MB Download

✓ Mentioned issues (3) ✓ Links (3)

Mentioned issues (3)

--	--	“(internal bug b/266179436 has log file and shows that Giraffe Canary 4 still has the NoClassDefFoundError for OCNamesInterner.)”
P2	Unable to debug androidTest-specific native code	“The attached project in b/268223043 (Google-internal-only bug) consistently reproduces the "Infinite Loader" issue for me (Giraffe Canary 2 a
--	--	“I'm resolving this bug and spinning of comment#14 into its own bug b/271119886 .”

Links (3)

“https://googleplex-android-review.git.corp.google.com/c/platform/tools...”
“https://fusion2.corp.google.com/invocations/a5d7f225-4ef7-482a-abc4-cc8de93d8e86/targets/%2F%2Ftools%2Fadt%2Fidea%2Fandroid-uitests:APKDebuggingProjTest;config=675d4cbfc071511... ’
“Android Studio Giraffe Canary 6 is available to download today: https://developer.android.com/studio/preview”

COMMENTS

em...@google.com <em...@google.com>

Reassigned to em...@google.com.

em...@google.com <em...@google.com>

Reassigned to jo...@google.com.

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

Doesn't repro with ToT on MacBook

jo...@google.com <jo...@google.com> #3

Doesn't repro on Giraffe Canary 01 on MacBook (still haven't gotten Canary 02 to work)

jo...@google.com <jo...@google.com> #4

Doesn't repro on Giraffe Canary 01 on Linux

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

Doesn't repro on Giraffe Canary 02 on Linux. It looks like this issue doesn't repeat reliably.

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

This stack from the log looks potentially relevant:

```
Caused by: java.lang.NoClassDefFoundError: com/jetbrains/cidr/lang/symbols/symtable/OCNamesInterner
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.InterningStringSerializer.read(InterningStringSerializer.kt:10)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.InterningStringSerializer.read(InterningStringSerializer.kt:8)
	at com.esotericsoftware.kryo.Kryo.readObjectOrNull(Kryo.java:730)
	at com.esotericsoftware.kryo.serializers.MapSerializer.read(MapSerializer.java:131)
	at com.esotericsoftware.kryo.serializers.MapSerializer.read(MapSerializer.java:21)
	at com.esotericsoftware.kryo.Kryo.readObject(Kryo.java:679)
	at com.esotericsoftware.kryo.serializers.ObjectField.read(ObjectField.java:106)
	at com.esotericsoftware.kryo.serializers.FieldSerializer.read(FieldSerializer.java:528)
	at com.esotericsoftware.kryo.Kryo.readObject(Kryo.java:657)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.FileSymbolTableSerializer.readObject(FileSymbolTableSerializer.java:229)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.FileSymbolTableSerializer.readMetaData(FileSymbolTableSerializer.java:249)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.SerializationServiceImpl.readMetaData(SerializationServiceImpl.java:364)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.SerializationServiceImpl.readTablesForFile(SerializationServiceImpl.java:279)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.SerializationServiceImpl.lambda$deserializeTables$2(SerializationServiceImpl.java:138)
	at com.intellij.openapi.application.impl.ApplicationImpl.runReadAction(ApplicationImpl.java:873)
	at com.intellij.openapi.application.ReadAction.compute(ReadAction.java:68)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.SerializationServiceImpl.lambda$deserializeTables$5(SerializationServiceImpl.java:133)
	at com.jetbrains.cidr.lang.symbols.symtable.serialization.SerializationServiceImpl.lambda$performAsynchronously$6(SerializationServiceImpl.java:236)
	at java.base/java.util.concurrent.ForkJoinTask$AdaptedRunnableAction.exec(ForkJoinTask.java:1375)
	at java.base/java.util.concurrent.ForkJoinTask.doExec(ForkJoinTask.java:373)
	at java.base/java.util.concurrent.ForkJoinPool$WorkQueue.topLevelExec(ForkJoinPool.java:1182)
	at java.base/java.util.concurrent.ForkJoinPool.scan(ForkJoinPool.java:1655)
	at java.base/java.util.concurrent.ForkJoinPool.runWorker(ForkJoinPool.java:1622)
	at java.base/java.util.concurrent.ForkJoinWorkerThread.run(ForkJoinWorkerThread.java:165)
Caused by: java.lang.ClassNotFoundException: com.jetbrains.cidr.lang.symbols.symtable.OCNamesInterner PluginClassLoader(plugin=PluginDescriptor(name=CIDR Debugger, id=com.intelli
path=~\Documents\Girafee\Canary2\android-studio-2022.3.1.2-windows\android-studio\plugins\cidr-debugger-plugin, version=223.7571.182.2231.9506613, package=null, isBundled=true),
	at com.intellij.ide.plugins.cl.PluginClassLoader.loadClass(PluginClassLoader.java:215)
	at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:520)
	... 24 more
```

jo...@google.com <jo...@google.com> #7

Marked as fixed.

Speaking with Emre we think this bug was resolved by:
https://googleplex-android-review.git.corp.google.com/c/platform/tools/idea/+/20976752
I'm marking this as resolved. Please reactivate if the issue reoccurs in Canary 3

at...@gmail.com <at...@gmail.com> #8

It still reproduces for me. I think the steps to reproduce is somehow different now.

Probably the way how you confirm as fixed is wrong. Once the editor tab is opened for the file, even if it's infinitely loading, the file will be shown if the project is closed and reopened.

But that does not mean the problem is fixed. If you close the editor tab for the .cpp file, close the project and reopen it, the problem is still there and the editor tab for the .cpp file is stuck at "

em...@google.com <em...@google.com> #9

Status: Assigned (reopened)

Can you please add full Android Studio version information and attach a new `idea.log` file?

em...@google.com <em...@google.com> #10

(internal bug b/266179436 has log file and shows that Giraffe Canary 4 still has the NoClassDefFoundError for OCNamesInterner.)

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

The exception reproduces when I hit the "Debug" button. I also see the `java.lang.AssertionError: Symbols must be loaded` errors without using the debugger.

...assigning back to jomof@

em...@google.com <em...@google.com> #12

The attached project in b/268223043 (Google-internal-only bug) consistently reproduces the "Infinite Loader" issue for me (Giraffe Canary 2 and Giraffe Canary 4) (Windows)

Message last modified on Feb 10, 2023 09:39AM

em...@google.com <em...@google.com> #13

I think I got to the bottom of the `NoClassDefFoundError`:

1. I was focusing on stack trace in comment#6, but I should have looked at the last line:
    - `PluginClassLoader(plugin=PluginDescriptor(name=CIDR Debugger...).`
2. This means the `OCNamesInterner` class was being searched by the `CIDR Debugger` plugin's class loader.
3. The class that triggered `OCNamesInterner` to be loaded is `InterningStringSerializer.` So, that class must have been loaded by the `CIDR Debugger` plugin classloader too. Verifica
    - `InterningStringSerializer` comes from `intellij.cidr.util.iml` module.
    - We embed `intellij.cidr.util.iml` into the `cidr-debugger-plugin.`
    - So, yes, that's why it's using `CIDR Debugger` plugin class loader.
4. Problem: `OCNamesInterner` is not inside the same plugin as `InterningStringSerializer`, i.e., `cidr-debugger-plugin.`

- `OCNamesInterner` is part of the `intellij.cidr.util.iml` module
- That module is embedded into the `cidr-base-plugin.` It used to be in `cidr-debugger-plugin`, but this was changed with `Commit 14347d7: CPP-30905 Return CidrUtilPlugin.x`

So, we are hitting the problem where the class loader of one plugin cannot load a class from inside another plugin.

Note:

- My repro attempts on `studio-main` did not work because I was building from IJ IDEA, which builds Android Studio with a single `PathClassLoader` class loader.
- I should have used the Bazel build, which builds Android Studio with `PluginClassLoader.`

Proposed fix:

- Move the `intellij.cidr.util.serialization` (and `intellij.cidr.util.ui` too?) module from the `cidr-debugger-plugin` to the `cidr-base-plugin.` The debugger plugin only us with CLion.

Message last modified on Feb 10, 2023 12:18PM

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

The proposed fix in comment#13 is submitted. It should be included in Giraffe Canary 6.

We have one more issue remaining, and that's about `clangd` being queried before indexing is finished, which causes an exception like this:

```
com.intellij.openapi.project.IndexNotReadyException: Please change caller according to com.intellij.openapi.project.IndexNotReadyException documentation
        at com.intellij.openapi.project.IndexNotReadyException.create(IndexNotReadyException.java:66)
        at com.intellij.util.indexing.FileBasedIndexImpl.handleDumbMode(FileBasedIndexImpl.java:841)
        at com.intellij.util.indexing.FileBasedIndexImpl.ensureUpToDate(FileBasedIndexImpl.java:789)
        at com.intellij.util.indexing.FileBasedIndexEx.processExceptions(FileBasedIndexEx.java:322)
        at com.intellij.util.indexing.FileBasedIndexEx.collectFileIdsContainingAllKeys(FileBasedIndexEx.java:615)
        at com.intellij.util.indexing.FileBasedIndexEx.processFilesContainingAllKeys(FileBasedIndexEx.java:455)
        at com.intellij.util.indexing.FileBasedIndexImpl.processFilesContainingAllKeys(FileBasedIndexImpl.java:1134)
        at com.intellij.util.indexing.FileBasedIndexEx.getFilesWithKey(FileBasedIndexEx.java:532)
        at com.intellij.find.ngrams.TrigramTextSearchService.processFilesWithText(TrigramTextSearchService.java:25)
        at com.jetbrains.cidr.lang.daemon.clang.clangd.settings.CppModulesStateImpl$ScanIndex.lambda$performImpl$0(CppModulesStateImpl.java:506)
        at com.jetbrains.cidr.lang.daemon.clang.clangd.settings.CppModulesStateImpl$ScanIndex.lambda$performImpl$2(CppModulesStateImpl.java:514)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$OTelMonitor.callWrapped(NonBlockingReadActionImpl.java:746)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$OTelMonitor$MonitoredComputation.call(NonBlockingReadActionImpl.java:778)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.insideReadAction(NonBlockingReadActionImpl.java:573)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.lambda$attemptComputation$3(NonBlockingReadActionImpl.java:537)
        at com.intellij.openapi.application.impl.ApplicationImpl.tryRunReadAction(ApplicationImpl.java:1086)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.lambda$runInReadActionWithWriteActionPriority$0(ProgressIndicatorUtils.java:71)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtilService.runActionAndCancelBeforeWrite(ProgressIndicatorUtilService.java:63)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.runActionAndCancelBeforeWrite(ProgressIndicatorUtils.java:128)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.lambda$runWithWriteActionPriority$1(ProgressIndicatorUtils.java:109)
        at com.intellij.openapi.progress.ProgressManager.lambda$runProcess$0(ProgressManager.java:68)
        at com.intellij.openapi.progress.impl.CoreProgressManager.lambda$runProcess$2(CoreProgressManager.java:188)
        at com.intellij.openapi.progress.impl.CoreProgressManager.lambda$executeProcessUnderProgress$13(CoreProgressManager.java:589)
        at com.intellij.openapi.progress.impl.CoreProgressManager.registerIndicatorAndRun(CoreProgressManager.java:664)
        at com.intellij.openapi.progress.impl.CoreProgressManager.computeUnderProgress(CoreProgressManager.java:620)
        at com.intellij.openapi.progress.impl.CoreProgressManager.executeProcessUnderProgress(CoreProgressManager.java:588)
        at com.intellij.openapi.progress.impl.ProgressManagerImpl.executeProcessUnderProgress(ProgressManagerImpl.java:60)
        at com.intellij.openapi.progress.impl.CoreProgressManager.runProcess(CoreProgressManager.java:175)
        at com.intellij.openapi.progress.ProgressManager.runProcess(ProgressManager.java:68)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.runWithWriteActionPriority(ProgressIndicatorUtils.java:106)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.runInReadActionWithWriteActionPriority(ProgressIndicatorUtils.java:71)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.attemptComputation(NonBlockingReadActionImpl.java:537)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.executeSynchronously(NonBlockingReadActionImpl.java:464)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl.executeSynchronously(NonBlockingReadActionImpl.java:218)
        at com.jetbrains.cidr.lang.daemon.clang.clangd.settings.CppModulesStateImpl$ScanIndex.performImpl(CppModulesStateImpl.java:514)
        at com.jetbrains.cidr.lang.daemon.clang.clangd.settings.CppModulesStateImpl$UpdateEvent.perform(CppModulesStateImpl.java:424)
        at com.jetbrains.cidr.lang.daemon.clang.clangd.settings.CppModulesStateImpl.lambda$new$2(CppModulesStateImpl.java:70)
        at com.intellij.util.concurrency.QueueProcessor.lambda$wrappingProcessor$0(QueueProcessor.java:80)
        at com.intellij.util.concurrency.QueueProcessor.runSafely(QueueProcessor.java:241)
        at com.intellij.util.concurrency.QueueProcessor.lambda$wrappingProcessor$1(QueueProcessor.java:80)
        at com.intellij.util.concurrency.QueueProcessor.lambda$startProcessing$3(QueueProcessor.java:215)
        at com.intellij.util.concurrency.QueueProcessor.runSafely(QueueProcessor.java:241)
        at com.intellij.util.concurrency.QueueProcessor.lambda$startProcessing$4(QueueProcessor.java:215)
        at com.intellij.openapi.application.impl.ApplicationImpl$1.run(ApplicationImpl.java:246)
        at java.base/java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:539)
        at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
        at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1136)
        at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635)
        at java.base/java.util.concurrent.Executors$PrivilegedThreadFactory$1$1.run(Executors.java:702)
        at java.base/java.util.concurrent.Executors$PrivilegedThreadFactory$1$1.run(Executors.java:699)
        at java.base/java.security.AccessController.doPrivileged(AccessController.java:399)
        at java.base/java.util.concurrent.Executors$PrivilegedThreadFactory$1.run(Executors.java:699)
        at java.base/java.lang.Thread.run(Thread.java:833)
```

jh...@themeetgroup.com <jh...@themeetgroup.com> #15

I'm also seeing what appears to be a different stack trace for the same symptom (opening a .cpp or .h file, showing perpetual "Loading.."):

```
2023-02-10 17:10:06,282 [1321370] SEVERE - #c.j.c.lang -        Symbol building is not allowed: 1676067006280 .
       Loaded: false
       Dumb:   false
       Previous activities:
       rebuildSymbols 1676065691915: START
rebuildSymbols 1676065892472: START
java.lang.Throwable:        Symbol building is not allowed: 1676067006280 .
       Loaded: false
       Dumb:   false
       Previous activities:
       rebuildSymbols 1676065691915: START
rebuildSymbols 1676065892472: START
        at com.intellij.openapi.diagnostic.Logger.error(Logger.java:202)
        at com.jetbrains.cidr.lang.symbols.symtable.building.OCBuildingActivityExecutionService.assertParsingAndSymbolBuildingAllowed(OCBuildingActivityEx
        at com.jetbrains.cidr.lang.symbols.symtable.FileSymbolTable.forFile(FileSymbolTable.java:228)
        at com.jetbrains.cidr.lang.symbols.symtable.FileSymbolTable.forFile(FileSymbolTable.java:220)
        at com.jetbrains.cidr.lang.psi.impl.OCFileImpl.getSymbolTable(OCFileImpl.java:252)
        at com.jetbrains.cidr.lang.psi.impl.OCFileImpl.processSymbolsRecursively(OCFileImpl.java:197)
        at com.jetbrains.cidr.lang.psi.impl.OCFileImpl.processSymbolsRecursively(OCFileImpl.java:185)
        at com.android.tools.ndk.jni.cosmetics.JniMangledFunctionNameFoldingBuilder.buildFoldRegions(JniMangledFunctionNameFoldingBuilder.kt:37)
        at com.intellij.lang.folding.LanguageFolding.buildFoldingDescriptorsNoPlaceholderCaching(LanguageFolding.java:89)
        at com.intellij.lang.folding.CompositeFoldingBuilder.buildFoldRegions(CompositeFoldingBuilder.java:36)
        at com.intellij.lang.folding.LanguageFolding.buildFoldingDescriptorsNoPlaceholderCaching(LanguageFolding.java:89)
        at com.intellij.lang.folding.LanguageFolding.buildFoldingDescriptors(LanguageFolding.java:72)
        at com.intellij.codeInsight.folding.impl.FoldingUpdate.getFoldingsFor(FoldingUpdate.java:272)
        at com.intellij.codeInsight.folding.impl.FoldingUpdate.getFoldingsFor(FoldingUpdate.java:238)
        at com.intellij.codeInsight.folding.impl.CodeFoldingManagerImpl.buildInitialFoldings(CodeFoldingManagerImpl.java:135)
        at com.intellij.openapi.fileEditor.impl.text.PsiAwareTextEditorImpl.lambda$loadEditorInBackground$0(PsiAwareTextEditorImpl.java:60)
        at com.intellij.openapi.fileEditor.impl.text.PsiAwareTextEditorImpl.catchingExceptions(PsiAwareTextEditorImpl.java:108)
        at com.intellij.openapi.fileEditor.impl.text.PsiAwareTextEditorImpl.loadEditorInBackground(PsiAwareTextEditorImpl.java:59)
        at com.intellij.openapi.fileEditor.impl.text.AsyncEditorLoader.scheduleLoading$lambda-1$lambda-0(AsyncEditorLoader.kt:115)
        at com.intellij.openapi.progress.impl.CoreProgressManager.computePrioritized(CoreProgressManager.java:774)
        at com.intellij.openapi.fileEditor.impl.text.AsyncEditorLoader.scheduleLoading$lambda-1(AsyncEditorLoader.kt:113)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$OTelMonitor.callWrapped(NonBlockingReadActionImpl.java:746)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$OTelMonitor$MonitoredComputation.call(NonBlockingReadActionImpl.java:778)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.insideReadAction(NonBlockingReadActionImpl.java:573)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.lambda$attemptComputation$3(NonBlockingReadActionImpl.java:537)
        at com.intellij.openapi.application.impl.ApplicationImpl.tryRunReadAction(ApplicationImpl.java:1086)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.lambda$runInReadActionWithWriteActionPriority$0(ProgressIndicatorUtils.java:71)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtilService.runActionAndCancelBeforeWrite(ProgressIndicatorUtilService.java:63)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.runActionAndCancelBeforeWrite(ProgressIndicatorUtils.java:128)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.lambda$runWithWriteActionPriority$1(ProgressIndicatorUtils.java:109)
        at com.intellij.openapi.progress.ProgressManager.lambda$runProcess$0(ProgressManager.java:68)
        at com.intellij.openapi.progress.impl.CoreProgressManager.lambda$runProcess$2(CoreProgressManager.java:190)
        at com.intellij.openapi.progress.impl.CoreProgressManager.lambda$executeProcessUnderProgress$13(CoreProgressManager.java:591)
        at com.intellij.openapi.progress.impl.CoreProgressManager.registerIndicatorAndRun(CoreProgressManager.java:666)
        at com.intellij.openapi.progress.impl.CoreProgressManager.computeUnderProgress(CoreProgressManager.java:622)
        at com.intellij.openapi.progress.impl.CoreProgressManager.executeProcessUnderProgress(CoreProgressManager.java:590)
        at com.intellij.openapi.progress.impl.ProgressManagerImpl.executeProcessUnderProgress(ProgressManagerImpl.java:60)
        at com.intellij.openapi.progress.impl.CoreProgressManager.runProcess(CoreProgressManager.java:177)
        at com.intellij.openapi.progress.ProgressManager.runProcess(ProgressManager.java:68)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.runWithWriteActionPriority(ProgressIndicatorUtils.java:106)
        at com.intellij.openapi.progress.util.ProgressIndicatorUtils.runInReadActionWithWriteActionPriority(ProgressIndicatorUtils.java:71)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.attemptComputation(NonBlockingReadActionImpl.java:537)
        at com.intellij.openapi.application.impl.NonBlockingReadActionImpl$Submission.lambda$transferToBgThread$1(NonBlockingReadActionImpl.java:444)
        at com.intellij.openapi.fileEditor.impl.text.AsyncEditorLoader$executor$1$execute$1.invokeSuspend(AsyncEditorLoader.kt:41)
        at kotlin.coroutines.jvm.internal.BaseContinuationImpl.resumeWith(ContinuationImpl.kt:33)
        at kotlinx.coroutines.DispatchedTask.run(DispatchedTask.kt:106)
        at kotlinx.coroutines.internal.LimitedDispatcher.run(LimitedDispatcher.kt:42)
        at kotlinx.coroutines.scheduling.TaskImpl.run(Tasks.kt:95)
        at kotlinx.coroutines.scheduling.CoroutineScheduler.runSafely(CoroutineScheduler.kt:570)
        at kotlinx.coroutines.scheduling.CoroutineScheduler$Worker.executeTask(CoroutineScheduler.kt:750)
        at kotlinx.coroutines.scheduling.CoroutineScheduler$Worker.runWorker(CoroutineScheduler.kt:677)
        at kotlinx.coroutines.scheduling.CoroutineScheduler$Worker.run(CoroutineScheduler.kt:664)
2023-02-10 17:10:06,283 [1321371] SEVERE - #c.j.c.lang - Android Studio Giraffe | 2022.3.1 Canary 4  Build #AI-223.7571.182.2231.9569140
2023-02-10 17:10:06,284 [1321372] SEVERE - #c.j.c.lang - JDK: 17.0.5; VM: OpenJDK 64-Bit Server VM; Vendor: JetBrains s.r.o.
2023-02-10 17:10:06,284 [1321372] SEVERE - #c.j.c.lang - OS: Mac OS X
```

Is this caused by the same issue, related, or different issue?

sm...@google.com <sm...@google.com> #16

Corresponding Ui that is failing because of the same cause: //tools/adt/idea/android-uitests:APKDebuggingProjTest

https://fusion2.corp.google.com/invocations/a5d7f225-4ef7-482a-abc4-cc8de93d8e86/targets/%2F%2Ftools%2Fadt%2Fidea%2Fandroid-uitests:APKDebuggingProjTest;config=675d4cbfc07

Because the symbols are not resolved. Debug option is not working.

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

Android Studio Giraffe Canary 6 is available to download today: https://developer.android.com/studio/preview

Can you try that version and let us know if the infinite loader issue is fixed, or still occurs?

jh...@themeetgroup.com <jh...@themeetgroup.com> #18

Giraffe Canary 6 works for me on a quick test. Files open, properly linking between java and native sources.

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

Marked as fixed.

I'm resolving this bug and spinning of comment#14 into its own bug b/271119886.