R8 -alwaysinline feature

WAI Bug P2 + Add Hotlist

STATUS UPDATE No update yet. Edit

DESCRIPTION ro...@gmail.com created issue #1 Jul 15, 2020 11:21PM

The R8 -alwaysinline feature does not seem to do anything. Is this feature released. I am using Android Studio 4.0 with all available updated applied. In my proguard-rules.pro file I have:

-alwaysinline class com.nbstech.nfcphone.UBNative

but no methods get inlined.

I have also tried:
-alwaysinline class com.nbstech.nfcphone.UBNative {
    String obfuscateDefaultString(int);
}

This method is not in-lined. It is a very short method.

Thanks for any advice,
Robin Ehrlich
rehrlich@nbstech.com

✓ Links (4) Hide all

Links (4)

"http://proguard-rules.pro"	ro...@ #1, ro...@ #3, ro...@ #4
"component: Android Public Tracker < https://issuetracker.google.com/components/326788 > > App Development > Android Studio > Build Tools > Shrinker (R8)"	ro...@ #3, ro...@ #4, ro...@ #7
"http://proguard.sourceforge.net/manual/examples.html#… "	ch...@ #6, ro...@ #7
"See also https://android.googlesource.com/platform/sdk/+/master/files/proguard-android-… ."	ch...@ #6, ro...@ #7

COMMENTS All comments ↓ Oldest first

ag...@google.com <ag...@google.com> #2 Jul 16, 2020 12:44AM

Reassigned to ag...@google.com.

Thanks for the report Robin. It should work, but it only works if the method is actually inlineable in the first place. One common reason why a method cannot be inlined is if your shrinker configuration does not contain `-allowaccessmodification`.

As an example:

```
class A {
  void foo(B b) {
    b.inlineMe();
  }
}

class B {
  private void cannotCallThisFromA() {
    ...
  }

  void inlineMe() {
    cannotCallThisFromA();
  }
}
```

If for some reason `cannotCallThisFromA` needs to stay in the output or is called from multiple places so we determine that inlining it is not worth it, we cannot inline `inlineMe` as that would leave a call in `A.foo` to `B.cannotCallThisFromA`.

So, we do honor `-alwaysinline`, but only if we can do so while generating valid code.

Can you provide more details of what you are seeing?

Message last modified on Jul 16, 2020 12:45AM

ro...@gmail.com <ro...@gmail.com> #3 Jul 16, 2020 02:56AM

Thank you very much for the quick response, but adding -allowaccessmodification, but it did not help. I have attached my proguard-rules.pro and the file I am trying to inline (UBNative.java). UBNative.java is in a library (.aar) file if that makes a difference.

I also tried:

-alwaysinline class com.nbstech.nfcphone.UBNative

without any methods and that failed also.

The reason for my trying to inline these functions is to further hide calls to the JNI code.

Thanks for your help,

Robin

From: buganizer-system@google.com <buganizer-system@google.com>
Sent: Wednesday, July 15, 2020 9:45 AM
To: b-system+-1433749314@google.com
Cc: robin.ehrlich@gmail.com
Subject: Re: Issue 161298405: R8 -alwaysinline feature

Replying to this email means your email address will be shared with the team that works on this product.

https://issuetracker.google.com/issues/161298405

Changed

<mailto:ag...@google.com> ag...@google.com added comment #2 <https://issuetracker.google.com/issues/161298405#comment2> :

Thanks for the report Robin. It should work, but it only works if the method is actually inlineable in the first place. One common reason why a method cannot be inlined is if your shrinker configuration does not contain -allowaccessmodification.

As an example:

```
class A {
  void foo(B b) {
    b.inlineMe();
  }
}

class B {
  private void cannotCallThisFromA() {
    ...
  }

  void inlineMe() {
    cannotCallThisFromA();
  }
}
```

If for some reason cannotCallThisFromA needs to stay in the output or is called from multiple places so we determine that inlining it is not worth it, we cannot inline inlineMe as that would leave a call in A.foo to B.cannotCallThisFromA.

So, we do honor -alwaysinline, but only if we can do so while generating valid code.

Can you provide more details of what you are seeing?

_____

Reference Info: 161298405 R8 -alwaysinline feature

component: Android Public Tracker <https://issuetracker.google.com/components/326788> > App Development > Android Studio > Build Tools > Shrinker (R8)
status: Assigned
reporter: robin.ehrlich@gmail.com <mailto:robin.ehrlich@gmail.com>
assignee: ag...@google.com <mailto:ag...@google.com>
cc: ag...@google.com <mailto:ag...@google.com> , r8...@googlegroups.com <mailto:r8...@googlegroups.com> , robin.ehrlich@gmail.com <mailto:robin.ehrlich@gmail.com>
type: Bug

priority: P2
severity: S2
retention: Component default

Generated by Google IssueTracker notification system

You're receiving this email because you are subscribed to updates on Google IssueTracker issue 161298405 <https://issuetracker.google.com/issues/161298405> where you have the roles: reporter, cc.
Unsubscribe from this issue. <https://issuetracker.google.com/issues/161298405?unsubscribe=true>

proguard-rules.pro
1.9 KB Download

UBNative.java
10 KB Download

ro...@gmail.com <ro...@gmail.com> #4 Jul 16, 2020 03:39AM

I should have also mentioned that proguard used to move these functions inline automatically.

Robin

From: Robin Ehrlich <robin.ehrlich@gmail.com>
Sent: Wednesday, July 15, 2020 11:57 AM
To: buganizer-system+326788+161298405@google.com; b-system+-1433749314@google.com
Subject: RE: Issue 161298405: R8 -alwaysinline feature

Thank you very much for the quick response, but adding -allowaccessmodification, but it did not help. I have attached my proguard-rules.pro and the file I am trying to inline (UBNative.java). UBNative.java is in a library (.aar) file if that makes a difference.

I also tried:

-alwaysinline class com.nbstech.nfcphone.UBNative

without any methods and that failed also.

The reason for my trying to inline these functions is to further hide calls to the JNI code.

Thanks for your help,

Robin

From: buganizer-system@google.com <mailto:buganizer-system@google.com> <buganizer-system@google.com <mailto:buganizer-system@google.com> >
Sent: Wednesday, July 15, 2020 9:45 AM
To: b-system+-1433749314@google.com <mailto:b-system+-1433749314@google.com>
Cc: robin.ehrlich@gmail.com <mailto:robin.ehrlich@gmail.com>
Subject: Re: Issue 161298405: R8 -alwaysinline feature

Replying to this email means your email address will be shared with the team that works on this product.

https://issuetracker.google.com/issues/161298405

Changed

<mailto:ag...@google.com> ag...@google.com added comment #2 <https://issuetracker.google.com/issues/161298405#comment2> :

Thanks for the report Robin. It should work, but it only works if the method is actually inlineable in the first place. One common reason why a method cannot be inlined is if your shrinker configuration does not contain -allowaccessmodification.

As an example:

class A {

```
  void foo(B b) {
    b.inlineMe();
  }
}

class B {
  private void cannotCallThisFromA() {
    ...
  }

  void inlineMe() {
    cannotCallThisFromA();
  }
}
```

If for some reason cannotCallThisFromA needs to stay in the output or is called from multiple places so we determine that inlining it is not worth it, we cannot inline inlineMe as that would leave a call in A.foo to B.cannotCallThisFromA.

So, we do honor -alwaysinline, but only if we can do so while generating valid code.

Can you provide more details of what you are seeing?

Reference Info: 161298405 R8 -alwaysinline feature

component: Android Public Tracker <https://issuetracker.google.com/components/326788> > App Development > Android Studio > Build Tools > Shrinker (R8)
status: Assigned
reporter: robin.ehrlich@gmail.com <mailto:robin.ehrlich@gmail.com>
assignee: ag...@google.com <mailto:ag...@google.com>
cc: ag...@google.com <mailto:ag...@google.com> , r8...@googlegroups.com <mailto:r8...@googlegroups.com> , robin.ehrlich@gmail.com <mailto:robin.ehrlich@gmail.com>
type: Bug
priority: P2
severity: S2
retention: Component default

Generated by Google IssueTracker notification system

You're receiving this email because you are subscribed to updates on Google IssueTracker issue 161298405 <https://issuetracker.google.com/issues/161298405> where you have the roles: reporter, cc.
Unsubscribe from this issue. <https://issuetracker.google.com/issues/161298405?unsubscribe=true>

---

ag...@google.com <ag...@google.com> #5 Jul 17, 2020 05:09PM

Reassigned to sg...@google.com.

Thanks for the additional information. We should have a look to see if calls to native methods ends up disabling this. Reassigning as I'm out on vacation for a couple of weeks starting today and I will not have time to dig in more today.

---

ch...@google.com <ch...@google.com> Aug 3, 2020 04:50PM

Reassigned to ch...@google.com.

---

ch...@google.com <ch...@google.com> #6 Aug 4, 2020 11:26PM

Status: Won't Fix (Intended Behavior)

Thanks for sharing. The reason why `UBNative.obfuscateDefaultString()` can't be inlined into other classes than `UBNative` is that it calls `UBNative.d()` which is private (thus inlining `UBNative.obfuscateDefaultString()` into another class would lead to an `IllegalAccessError`).

The reason why `UBNative.d()` is not made private although you are building with `-allowaccessmodification` is that it is kept by a `-keep` rule. In particular, this method is native, so I would expect it to be kept by the following rule from proguard-android.txt/proguard-android-optimize.txt:

```
# For native methods, see http://proguard.sourceforge.net/manual/examples.html#native
-keepclasseswithmembernames class * {
    native <methods>;
}
```

See also https://android.googlesource.com/platform/sdk/+/master/files/proguard-android-optimize.txt.

If a method is kept, its visibility could potentially be accessed by reflection, and therefore R8 is not allowed to make it public.

I can think of two solutions to solve your problem:

1. You can make the native methods public (it might also suffice to make them package-private, depending on where they are accessed from).
2. You can change the above keep rule to:

```
-keepclasseswithmembernames,allowaccessmodification class * {
    native <methods>;
```

```
}
```

This explicitly tells R8 that it is allowed to make the native methods public although they are kept. However, this is a fairly recent addition to R8, so this only works if you are using R8 version 2.1 or higher.

If the above proposals doesn't solve the inlining problem please don't hesitate to reopen this issue!

---

ro...@gmail.com <ro...@gmail.com> #7 Aug 7, 2020 11:17PM

Thank you very much for your help.

Using suggestion #1, making the native functions package-private solved the problem.

I have great respect for google that you were willing to spend the time helping me resolve this issue.

From: buganizer-system@google.com <buganizer-system@google.com>
Sent: Tuesday, August 04, 2020 8:27 AM
To: b-system+-1433749314@google.com
Cc: robin.ehrlich@gmail.com
Subject: Re: Issue 161298405: R8 -alwaysinline feature

Replying to this email means your email address will be shared with the team that works on this product.

https://issuetracker.google.com/issues/161298405

Changed
status: Assigned → Intended Behavior

<mailto:ch...@google.com> ch...@google.com added comment #6 <https://issuetracker.google.com/issues/161298405#comment6> :

Thanks for sharing. The reason why UBNative.obfuscateDefaultString() can't be inlined into other classes than UBNative is that it calls UBNative.d() which is private (thus inlining UBNative.obfuscateDefaultString() into another class would lead to an IllegalAccessError).

The reason why UBNative.d() is not made private although you are building with -allowaccessmodification is that it is kept by a -keep rule. In particular, this method is native, so I would expect it to be kept by the following rule from proguard-android.txt/proguard-android-optimize.txt:

```
# For native methods, see http://proguard.sourceforge.net/manual/examples.html#native
-keepclasseswithmembernames class * {
    native <methods>;
}
```

See also https://android.googlesource.com/platform/sdk/+/master/files/proguard-android-optimize.txt.

If a method is kept, its visibility could potentially be accessed by reflection, and therefore R8 is not allowed to make it public.

I can think of two solutions to solve your problem:

1. You can make the native methods public (it might also suffice to make them package-private, depending on where they are accessed from).
2. You can change the above keep rule to:

```
3. -keepclasseswithmembernames,allowaccessmodification class * {
4.     native <methods>;
5. }
```

This explicitly tells R8 that it is allowed to make the native methods public although they are kept. However, this is a fairly recent addition to R8, so this only works if you are using R8 version 2.1 or higher.

If the above proposals doesn't solve the inlining problem please don't hesitate to reopen this issue!

Reference Info: 161298405 R8 -alwaysinline feature

component: Android Public Tracker <https://issuetracker.google.com/components/326788> > App Development > Android Studio > Build Tools > Shrinker (R8)
status: Intended Behavior
reporter: robin.ehrlich@gmail.com <mailto:robin.ehrlich@gmail.com>
assignee: ch...@google.com <mailto:ch...@google.com>
cc: ag...@google.com <mailto:ag...@google.com> , r8...@googlegroups.com <mailto:r8...@googlegroups.com> , robin.ehrlich@gmail.com <mailto:robin.ehrlich@gmail.com>
type: Bug
priority: P2
severity: S2
retention: Component default