

Forum:[Hack Forums](#)

Thread: [Decompilation advice needed](#)

Aussie - January 23, 2017, 12:26 am

Hypothetically, I decompile a runnable jar using JD-GUI, and if I wished to use it.

How would I know the versions of the libraries it uses?

For example, if it uses apache commons and google commons, how can I tell the EXACT version?

GenericSkid - January 23, 2017, 8:56 am

You can't just by decompiling. The only giveaways are the methods and fields used in the library.

3cli??pse - January 23, 2017, 9:50 am

(Jan 23, 2017 01:26 AM)n31ln3t Wrote: [\[CITE\]](#) Hypothetically, I decompile a runnable jar using JD-GUI, and if I wished to use it.

How would I know the versions of the libraries it uses?

For example, if it uses apache commons and google commons, how can I tell the EXACT version?

The only way to tell is by whatever one is bundled with the library, or what's on the classpath. Your bet bet would be to look at each libraries manifest and see if they've included the version there

(Jan 23, 2017 09:56 AM)Lord Pankake Wrote: [\[CITE\]](#) You can't just by decompiling. The only giveaways are the methods and fields used in the library.Ignore this guy, he's retarded.

GenericSkid - January 23, 2017, 10:47 am

(Jan 23, 2017 10:50 AM)Wiz0rd Wrote: [\[CITE\]](#) The only way to tell is by whatever one is bundled with the library, or what's on the classpath. Your bet bet would be to look at each libraries manifest and see if they've included the version there

What if the library isn't bundled? OP said nothing about having the libraries, just the application that uses them.

Your solution only works if he has the libraries. Depending on how Op obtained the file he may or may not have them.

3cli??pse - January 23, 2017, 11:11 am

(Jan 23, 2017 11:47 AM)Lord Pankake Wrote: [\[CITE\]](#) What if the library isn't bundled? OP said nothing about having the libraries, just the application that uses them.

Your solution only works if he has the libraries. Depending on how Op obtained the file he may or may not have them.

Looking at what fields and methods are referenced is literally retarded, and you can't base your decision off that. The only way to tell what version of an API is being used it to fucking look at the one that came with it ?

For fields: none should be referenced at all, as jvm folds all constants references at bytecode level, and no decent api publicly exposes member variables

For methods: They should generally always be there, even if underlying code has changed, it's the purposes of writing getters/setters in the first place.

As usual you don't think though what you're talking about, and make yourself seem god damn retarded Pankake.

clrlslwi - January 23, 2017, 11:15 am

(Jan 23, 2017 12:11 PM)Wiz0rd Wrote: [\[CITE\]](#) Looking at what fields and methods are referenced is literally retarded, and you can't base your decision off that. The only way to tell what version of an API is being used it to fucking look at the one that came with it ?

For fields: none should be referenced at all, as jvm folds all constants references at bytecode level, and no decent api publicly exposes member variables

For methods: They should generally always be there, even if underlying code has changed, it's the purposes of writing getters/setters in the first place.

As usual you don't think though what you're talking about, and make yourself seem god damn retarded Pankake.

Ignore this retard, his college didn't even teach him about NoSQL ?

You need to inspect the code written, and decide how retarded it is. More retarded = older libraries

GenericSkid - January 23, 2017, 11:29 am

(Jan 23, 2017 12:11 PM)Wiz0rd Wrote: [\[CITE\]](#) Looking at what fields and methods are referenced is literally retarded, and you can't base your decision off that. The only way to tell what version of an API is being used it to fucking look at the one that came with it ?

Some libraries add/remove/deprecate methods and classes in between versions every now and then and based on that information you can ballpark what version of the library is used if you don't have direct access to the library itself.

(Jan 23, 2017 12:15 PM)Demmonic Wrote: [\[CITE\]](#) Ignore this retard
Pls no use of the "R-word" it offend me and my kind.

(Jan 23, 2017 12:15 PM)Demmonic Wrote: [\[CITE\]](#) You need to inspect the code written, and decide how retarded it is. More retarded = older libraries

Perfect solution ?

3cli??pse - January 23, 2017, 11:36 am

Fuck it, just use the newest lib, shit should work no matter what. And no API removes methods, like ever. Only get marked at deprecated. And in the off chance something was removed, just go back one major revision and try again.

clrlslwi - January 23, 2017, 11:37 am

(Jan 23, 2017 12:29 PM)Lord Pankake Wrote: [\[CITE\]](#) Perfect solution ?

My advice is solid, bro. R words don't use build tools, meaning they never get updated cus they manually include them, and are too lazy to update them.

Aussie - January 23, 2017, 4:51 pm

Haha, I switched to maven, and I set the library at the highest, and kept downgrading until all the errors went poof! All done without the use of the 'R-word'

Thanks fellas, but I have new issues now. Unfortunately