

Good question, I don't know. All the method implementations are just copied verbatim from IAGDB. That line was from a commit made by Jakub in July last year. Maybe we should add it? Not sure why it was commented out, and if we can maybe safely add it back.

Add a line note





thobe added a note Apr 20, 2015

Owner

I can't quite see this flag serving a purpose. It is only assigned in the constructor, so it could be final. But the only cases when that assignment wouldn't take place are cases when the object wouldn't be published (because of exceptions in the constructor) so it does in fact serve no purpose.

Perhaps it should be traded for a flag that helps making shutdown idempotent?

rickardoberg added a note Apr 20, 2015

Collaborator

This is another one that was simply copied. It is not set in the constructor though, but rather the init() method. In 3.0 I hope we can change this to be constructor though, which would be better than the init() method.

But replacing it with shutdown flag is definitely an option.

thobe added a note Apr 20, 2015

Owner

No. I'm silly. that's not a constructor, that's a method called <u>init</u> ... I'm still wondering if shutdown() shouldn't set this field to false.

Add a line note

thobe commented on the diff Apr 20, 2015

...rg/neo4j/kernel/impl/factory/GraphDatabaseFacade.java

View full changes





thobe added a note Apr 20, 2015

Owner

These module fields doesn't appear to be used (except for platform module which is used for accessing the config , but that could be extracted to a separate field in the init method just like the other components of these objects).

I'd propose removing these fields from here. The module types still serve a purpose, as vectors of their respective fields, but after that purpose have been served, I don't see the point of keeping those objects around.



rickardoberg added a note Apr 20, 2015

Collaborator

The main reason I left them is if we want to have tests access them instead of using DependencyResolver. In general it would be nice if we could cut down on DependencyResolver usage, as it is a bit overused as it is. With the module fields it will be much easier to do "Find usages" when we make refactorings of services and whatnot. Makes sense?



rickardoberg added a note Apr 20, 2015

Collaborator

We could also opt for removing all the other fields, and have accesses go through module refs, as there should be no performance overhead for doing so.



thobe added a note Apr 20, 2015

Owner

Makes sense. I didn't notice that they were public.

We could have all access go through the module refs, although I do think there would be a (tiny) performance overhead of doing so, since it is one additional indirection.



lutovich added a note Apr 23, 2015

Collaborator

Imho public fields are very bad and replacing hackish DependencyResolver (which we at least have control over) with those is not a good idea.



rickardoberg added a note Apr 24, 2015

Collaborator

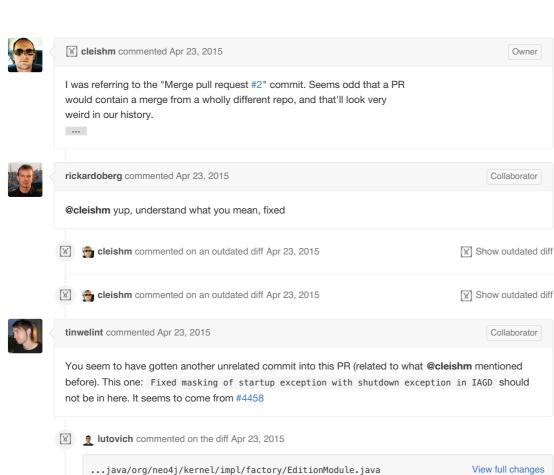
The reason is because with DependencyResolver it is non-trivial to know what services are used by what code. With the public fields we can do "Find usages" during refactorings. Making them final is a good point though.

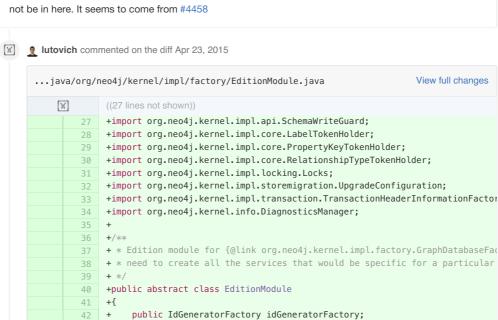
Add a line note

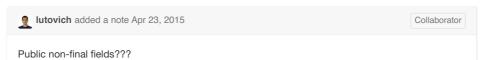


cleishm commented Apr 23, 2015

Squash?

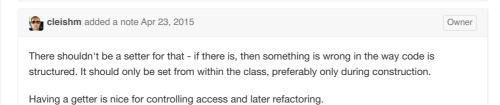






tinwelint added a note Apr 23, 2015 Collaborator

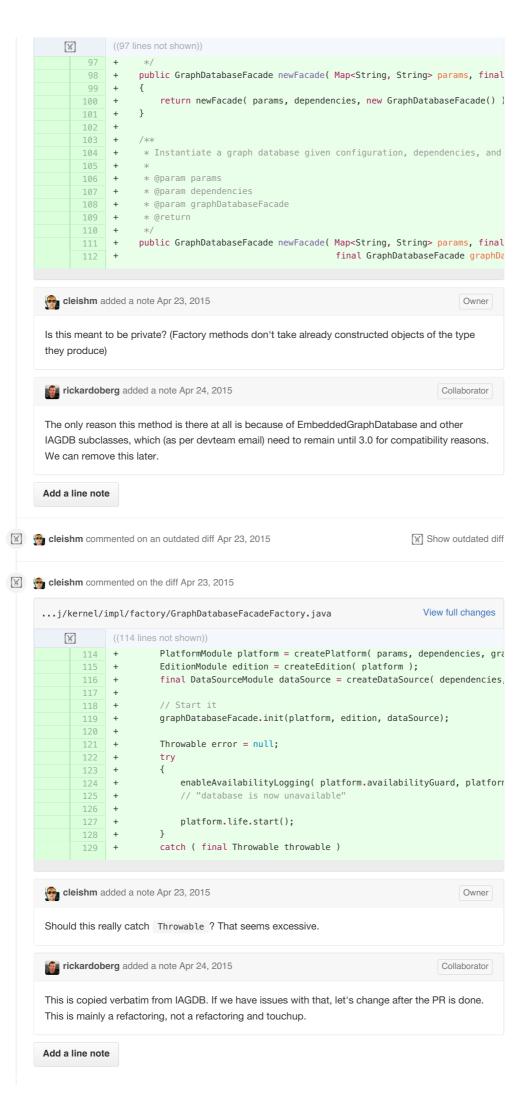
I don't think that's as bad as it looks, since that's internal classes. Compare that to a setter and a getter which effectively is the same thing. The difference is that you can control access to it, although since this is internal stuff we can change this at any time and the compiler will help us complete such a change.

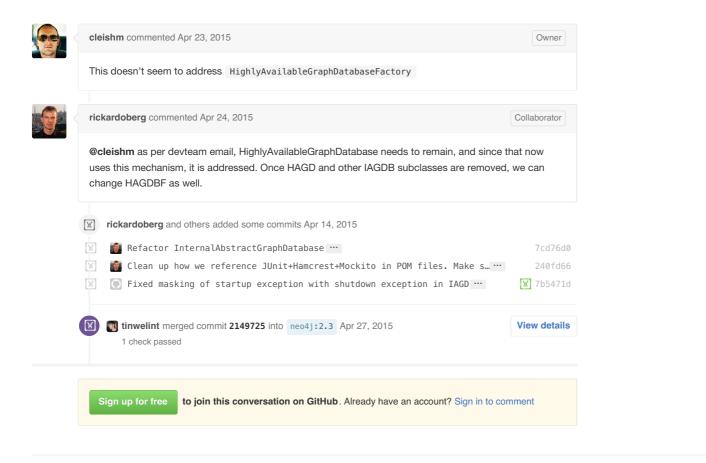


rickardoberg added a note Apr 24, 2015

Collaborator

I can add final, that's fine. Adding getters wouldn't add anything beyond that. Add a line note 👿 🤦 lutovich commented on an outdated diff Apr 23, 2015 tinwelint commented Apr 23, 2015 Personally I think the amount of introduced checkstyle violations is a bit on the high side. How can we improve this moving further? I'm not militant about fixing those before merge, but it'd make me happier. immelint commented on an outdated diff Apr 23, 2015 Show outdated diff implication tinwelint commented on an outdated diff Apr 23, 2015 Show outdated diff implication with the time of the time of the time in the time of time of the time of time of the time of t ...a/org/neo4j/kernel/impl/factory/DataSourceModule.java View full changes M ((316 lines not shown)) protected NodeProxy.NodeActions createNodeActions(final GraphDatabaseSet 316 318 319 return new NodeProxy.NodeActions() 320 321 + 322 + @Override public Statement statement() 324 return threadToStatementContextBridge.get(); 326 @Override 328 public GraphDatabaseService getGraphDatabase() 330 // TODO This should be wrapped as well Collaborator tinwelint added a note Apr 23, 2015 Should it be wrapped or is it an old comment? rickardoberg added a note Apr 24, 2015 Collaborator Most of this is copied verbatim from IAGDB. If we have issues with IAGDB code, then lets deal with that after this is merged. Add a line note Cleishm commented on the diff Apr 23, 2015 $\dots j/kernel/impl/factory/GraphDatabaseFacadeFactory.java$ View full changes





© 2015 GitHub,

Inc. Terms Privacy Security Contact

W

Status API Training Shop Blog About Help