Runtime Infrastructure - Feature #28

version the P2J code and/or jars

09/15/2011 12:00 PM - Greg Shah

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Hynek Cihlar	**% Done:**	100%
Category:		**Estimated time:**	12.00 hours
Target version:	Deployment and Management Improvements		
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Related to FWD - Feature #3063: Determining P2J version	**Closed**

History

#1 - 09/18/2011 05:35 PM - Redmine Admin

@45187 - Status changed from WIP to Hold

#2 - 04/12/2012 11:23 AM - Greg Shah

Imported from JPRM on 2012-04-12 11:23:23.535:

```
TASKID      = 6096
PROJECTID   = 125
STATUS      = 3 (Hold)
DESCRIPTION = version the P2J code and/or jars
OWNER       = GES
ASSIGNEE    =
BILLABLE    = false
PRIORITY    = 5 (Normal)
PHASE       = 31 (Development)
COMPONENT   = 0
ESTEFFORT   = 0.000000
ESTSTART    = 2011-09-15
ESTSTOP     =
ACTSTOP     =
CASENUM     =
VENDORID    =
COMMENT     = ''
LASTWIP     = 2011-09-18
```

#3 - 10/31/2012 11:30 AM - Greg Shah

- Estimated time set to 12.00

- Target version set to Deployment and Management Improvements

#4 - 11/16/2016 01:06 PM - Greg Shah

- Target version changed from Deployment and Management Improvements to Deployment and Management Improvements

#5 - 01/11/2017 01:59 PM - Greg Shah

- Status changed from New to WIP

- Project changed from Core Development to Runtime Infrastructure

- Assignee set to Hynek Cihlar

You can include changes for this into 2699a.

Please see #3063 for some useful discussion.

#6 - 01/11/2017 03:42 PM - Hynek Cihlar

According to #3063-3 the idea is to build the version from the following components.

- major version
- minor version
- release
- repo (e.g. p2j for the main trunk and branches or optionally this may be a custom build line for a specific customer)
- branch name
- branch revision

My only concern is that our branching process relies on rebasing. When a branch is rebased the revision numbers are increased for the whole set of rebased revisions. We would have to make sure not to ever rebase the release branch to get the revision numbers stable.

#7 - 01/12/2017 04:48 AM - Greg Shah

> We would have to make sure not to ever rebase the release branch to get the revision numbers stable.

I think this is a safe idea. Our entire development process is designed to be linear and ever growing/never shrinking.

#8 - 01/12/2017 03:55 PM - Hynek Cihlar

The initial version of build target versioning committed to 2699a. The version string is added to the jars' manifests as suggested in the previous notes.

There is a new file version.properties in the root project dir. The file defines the properties major, minor, release and repo. These properties are used to build up the final version string.

There are two version formats, a non-release and release, <major>.<minor>.<release>-SNAPSHOT_<repo> and <major>.<minor>.<release>_<repo>_<branch>_<revno> respectively. The idea behind the SNAPSHOT version is (1) to provide version scheme for devel builds and (2) to avoid the relatively lengthy resolution of branch name and revision number for devel builds.

Release format is triggered when the property release is defined for the gradle build (gradlew all -P release).

#9 - 01/12/2017 05:30 PM - Hynek Cihlar

I have committed a change of version scheme to make it compatible with existing schemes of Mavem, Ivy, etc.

The new version scheme is <major>.<minor>.<release>-SNAPSHOT-<repo> and <major>.<minor>.<release>-<repo>_<branch>_<revno>.

#10 - 01/18/2017 07:37 AM - Greg Shah

The problem with the SNAPSHOT approach is that it is very likely that copies taken from development branches will be in use by customers. This is the very reason we need the full repo-branch-revno information.

I know you did this to save time in common dev build situations. How much time is saved? If it is significant, then we can cache the repo and branch in a properties file that is built the first gradle run after check out. The bzr revno is fast (and changes often) so that can be read dynamically.

#11 - 01/18/2017 07:39 AM - Greg Shah

Please make changes to the server startup process such that the first entry written to the log is the version string ("FWD v<version_string_from_manifest> Server starting initialization.").

#12 - 01/18/2017 07:43 AM - Greg Shah

BTW, the repo should not be hard coded as p2j in the version.properties. We have customer-specific repositories because we provide custom build lines. For example, the customer for the ChUI regression testing application has a custom build of P2J that is based off trunk 10680, but has only a subset of the subsequent changes applied. That is what they use in production. Their customer name is the repo name and needs to be read from the checkout location (and cached) so that it is correct.

#13 - 01/18/2017 07:54 AM - Hynek Cihlar

Greg Shah wrote:

> The problem with the SNAPSHOT approach is that it is very likely that copies taken from development branches will be in use by customers. This is the very reason we need the full repo-branch-revno information.
>
> I know you did this to save time in common dev build situations. How much time is saved?

In my local dev environment it is usually between 3-5 seconds.

> If it is significant, then we can cache the repo and branch in a properties file that is built the first gradle run after check out. The bzr revno is fast (and changes often) so that can be read dynamically.

Another option would be to fetch the branch name only when the jar manifests (that hold the final version string) are not already built. That is, an initial all/jar task would query the name but subsequent jar task invocations not. Only the tasks that directly or indirectly cause the build dir removal (clean, all) would again cause the branch name query.

Yes, revno is no problem, that seems to be resolved without a network roundtrip.

Also I haven't solved the case when FWD is built outside of the bzr repo, for example when an open-source developer downloads a snapshot. What

should be the version string in this case?

#14 - 01/18/2017 07:59 AM - Greg Shah

> Another option would be to fetch the branch name only when the jar manifests (that hold the final version string) are not already built. That is, an initial all/jar task would query the name but subsequent jar task invocations not. Only the tasks that directly or indirectly cause the build dir removal (clean, all) would again cause the branch name query.

That is OK. But it seems like we still have to rebuild the manifest with each build because the bzr revno may have changed. That is why I thought that we should cache the details in a props file. The clean step can remove it and allow the re-query of the details.

#15 - 01/18/2017 08:01 AM - Hynek Cihlar

Greg Shah wrote:

> > Another option would be to fetch the branch name only when the jar manifests (that hold the final version string) are not already built. That is, an initial all/jar task would query the name but subsequent jar task invocations not. Only the tasks that directly or indirectly cause the build dir removal (clean, all) would again cause the branch name query.
>
> That is OK. But it seems like we still have to rebuild the manifest with each build because the bzr revno may have changed. That is why I thought that we should cache the details in a props file. The clean step can remove it and allow the re-query of the details.

You are right, the cached details (in a prop file) will be needed.

#16 - 01/18/2017 08:01 AM - Greg Shah

> Also I haven't solved the case when FWD is built outside of the bzr repo, for example when an open-source developer downloads a snapshot. What should be the version string in this case?

Good question. I think we want to have a props file with the details of the branch where the snapshot was taken. And we should add something that highlights the result as a snapshot like appending _snapshot to the version string.

#17 - 01/18/2017 08:36 AM - Greg Shah

Also, please add a Version class in the main package. When invoked from the command line (no parameters needed), it should output the version string "FWD v<version_string>".

The idea is that this allows someone with the jar to display the version without:

- running the server
- unpacking the manifest and looking inside

#18 - 01/18/2017 07:50 PM - Hynek Cihlar

2699a rev 11154 resolves the open issues above. I have simplified the implementation a bit. All the bzr information (repo name, branch and revision number) is fetched such that no network round trip is needed (assuming standard checkout bzr format) and so the caching logic could have been removed. Also I added format property to version.properties, this is to support other exotic use cases like building FWD from a file location not representing a valid bzr checkout.

I also added the com.goldencode.p2j.Version class with main() entry point printing FWD version string to the standard output. And added a log entry with FWD version on server startup.

#19 - 01/19/2017 07:33 AM - Greg Shah

Code Review Task Branch 2699a Revision 11154

Nice work. Some minor changes:

1. In Version.java, please split the if (!classPath.startsWith("jar")) { closing brace to the next line.

2. The changes in Utils seem unnecessary.

3. Should we comment out the format entry in version.properties? I'd prefer that the default came from the build.gradle and that this is just used optionally.

4. Please add Version to the p2jpl.jar.

5. Please add a user-defined function to p2jpl.jar (and the proper registration for it) so that we can check the version of the p2jpl.jar using SQL. This will allow us to remotely check the version number of the currently installed p2jpl.jar which could be on a separate DB server. Mismatches can be detected early, avoiding some issues.

Eric: It seems to make sense that we would also read this p2jpl.jar value in server startup and output this value to the server log. This would also be a chance to detect if the p2jpl.jar was missing entirely and report that.

#20 - 01/19/2017 09:12 AM - Hynek Cihlar

Greg Shah wrote:

> 1. In Version.java, please split the if (!classPath.startsWith("jar")) { closing brace to the next line.
>
> 2. The changes in Utils seem unnecessary.
>
> 3. Should we comment out the format entry in version.properties? I'd prefer that the default came from the build.gradle and that this is just used optionally.

1-3 resolved in 2699a revision 11155.

> 4. Please add Version to the p2jpl.jar.

Version was already included through the com/goldencode/p2j/* inclusion in build.xml.

> 5. Please add a user-defined function to p2jpl.jar (and the proper registration for it) so that we can check the version of the p2jpl.jar using SQL. This will allow us to remotely check the version number of the currently installed p2jpl.jar which could be on a separate DB server. Mismatches can be detected early, avoiding some issues.

User-defined function added in 2699a rev 11155, though I haven't tested it yet. Do we need version reporting for the C# binding, too?

#21 - 01/19/2017 09:43 AM - Greg Shah

> Do we need version reporting for the C# binding, too?

Yes, please.

#22 - 01/19/2017 09:49 AM - Greg Shah

Code Review Task Branch 2699a Revision 11155

It looks good except the pl/p2jpl.ddr needs to be updated to register the function.

Eric will respond on the C# side.

#23 - 01/19/2017 09:50 AM - Greg Shah

FYI, in pl/p2jpl.ddl there are two sections to update (install and uninstall).

#24 - 01/19/2017 09:51 AM - Hynek Cihlar

Greg Shah wrote:

> Do we need version reporting for the C# binding, too?

Yes, please.

Is there any documentation for this, build and installation instructions? Do I assume correctly that I will need a Windows box to be able to test the changes?

#25 - 01/19/2017 09:53 AM - Greg Shah

> Do I assume correctly that I will need a Windows box to be able to test the changes?

Eric will handle your other questions. Don't worry about testing the C# changes. We have some other issues to resolve there anyway and we don't have the time to get all of that worked out.

#26 - 01/19/2017 10:00 AM - Eric Faulhaber

Hynek Cihlar wrote:

> Greg Shah wrote:
>
> > Do we need version reporting for the C# binding, too?

> Yes, please.

> Is there any documentation for this, build and installation instructions? Do I assume correctly that I will need a Windows box to be able to test the changes?

Ovidiu, I know you documented this, but I can't find the latest instructions at the moment. I found #2143-24, but I think this is not the most recent. Can you point Hynek at the latest, please?

#27 - 01/19/2017 10:17 AM - Ovidiu Maxiniuc

> Ovidiu, I know you documented this, but I can't find the latest instructions at the moment. I found #2143-24, but I think this is not the most recent. Can you point Hynek at the latest, please?

Hynek,
Please see the #2332, note 1 and 6. I tried to keep the list complete, but the time allocated to Windows was little. Let me know if you encounter any issues. Eventually I will help you test on my Windows machine.

#28 - 01/19/2017 10:21 AM - Greg Shah

My intention with the C# stuff is to code it in the project but NOT to test it right now (unless OM can test it quickly). I don't want to spend the time on all the setup and so forth, which can take some time.

Ovidiu: Could you point Hynek to the files that need to be edited to expose a new C# wrapper function that calls the static Version.getFWDVersion() method? If there is anything to do to encode the registration of that function, let mention that too.

#29 - 01/19/2017 10:23 AM - Eric Faulhaber

Greg Shah wrote:

> Eric: It seems to make sense that we would also read this p2jpl.jar value in server startup and output this value to the server log. This would also be a chance to detect if the p2jpl.jar was missing entirely and report that.

Please add a package-private, static method to Persistence called reportUdfVersion. It should check out/in a database connection like we do in Persistence.queryIndexData. The SQL to get the version will just be select version() (or whatever your new version check UDF is called).

The call to Persistence.reportUdfVersion should be invoked in DatabaseManager.registerDatabase, just before the call to Persistence.queryAllIndexData(database, schema) (line 2353). We probably only want this to be written to the log if the bootstrap variable in DatabaseManager.registerDatabase is true, because this method also gets called for temporarily connected, remote databases. I think we would only want the message written at startup of the server which is authoritative for that database.

#30 - 01/19/2017 10:26 AM - Hynek Cihlar

I have added the new user function getFWDVersion() to the ddr file as well as to the C# sources. Ovidiu/Greg, please review.

The call to getFWDVersion() eventually ends up in com.goldencode.p2j.Version.getFWDVersion() which loads the manifest file using a technique which may not port well to IKVM. Ovidiu, if you find out that the new user function doesn't return a valid version string I will have to come up with a different, more portable way to store/load the version string.

#31 - 01/19/2017 10:41 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> My intention with the C# stuff is to code it in the project but NOT to test it right now (unless OM can test it quickly). I don't want to spend the time on all the setup and so forth, which can take some time.
>
> Ovidiu: Could you point Hynek to the files that need to be edited to expose a new C# wrapper function that calls the static Version.getFWDVersion() method? If there is anything to do to encode the registration of that function, let mention that too.

Hynek,
At this moment there are 3 C# source files that map the functions and operators used in queries. They are used only in server-side database processing. However, they do not call java code, instead they call other assemblies converted by ikvm from our java compiled code. Adding a new function is straightforward, just append it to Functions.cs (this seems the most likely location) using the syntax C# (it is similar to java): add a new body and return by calling the appropriate static method from FWD. I will do the testing, it I will compile/reconvert the code (so you don't have to set up a Windows station or even install/configure mono/ikvm).

BTW, the src/com/goldencode/p2j/persist/pl/AssemblyInfo.cs has some figures/strings you just need to update. I am not aware at this moment if the C# allows the assembly code to extract these, so we could return them directly, right from C# native code (but it should be possible I guess).

#32 - 01/19/2017 10:47 AM - Eric Faulhaber

Code review 2699a/11156:

Hynek, I'm only looking at the persistence-related changes.

Please fix a minor regression in p2jpl.ddr: in the drop functions section, getPrecisionScale was changed to getPrecisionSale.

The declaration of getFWDVersion in p2jpl.ddr needs to return text/java.lang.String instead of boolean/java.lang.Boolean. Using Common.fromBoolean in Functions.cs looks like a similar problem.

#33 - 01/19/2017 10:56 AM - Hynek Cihlar

Eric Faulhaber wrote:

> Code review 2699a/11156:
>
> Hynek, I'm only looking at the persistence-related changes.
>
> Please fix a minor regression in p2jpl.ddr: in the drop functions section, getPrecisionScale was changed to getPrecisionSale.
>
> The declaration of getFWDVersion in p2jpl.ddr needs to return text/java.lang.String instead of boolean/java.lang.Boolean. Using Common.fromBoolean in Functions.cs looks like a similar problem.

Eric, thanks for catching this braino. Fixed in 2699a revision 11157.

#34 - 01/19/2017 11:14 AM - Greg Shah

Code Review Task Branch 2699a Revision 11157

The changes look fine to me.

Ovidiu: Does the Functions.cs method getFWDVersion() need an underscore at the end of the name like getFWDVersion_()? I ask because the other methods in that file have this (and more if there are parameters).

#35 - 01/19/2017 11:21 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> Code Review Task Branch 2699a Revision 11157
>
> The changes look fine to me.
>
> Ovidiu: Does the Functions.cs method getFWDVersion() need an underscore at the end of the name like getFWDVersion_()? I ask because the other methods in that file have this (and more if there are parameters).

Yes. This is required by the naïve decoration implementation that blindly inserts it to delimit the signature.

#36 - 01/19/2017 12:19 PM - Ovidiu Maxiniuc

Just an idea:
wouldn't it be useful to have the version of the FWD that did the conversion written somewhere in generated jar?

Normally the same revision should be used for both conversion and runtime, but this is not mandatory and also this could be a solution to test the compatibility at the start-up of the server. I think the TRPL can update the project's jar manifest every time the project is reconverted, and from there it can be read by runtime and reported.

#37 - 01/19/2017 01:03 PM - Hynek Cihlar

Ovidiu Maxiniuc wrote:

> Ovidiu: Does the Functions.cs method getFWDVersion() need an underscore at the end of the name like getFWDVersion_()? I ask because the other methods in that file have this (and more if there are parameters).

> Yes. This is required by the naïve decoration implementation that blindly inserts it to delimit the signature.

The missing underscore is fixed in 2699a revision 11158.

#38 - 01/19/2017 01:07 PM - Hynek Cihlar

Ovidiu Maxiniuc wrote:

> I will do the testing, it I will compile/reconvert the code (so you don't have to set up a Windows station or even install/configure mono/ikvm).

Great, thanks!

> BTW, the src/com/goldencode/p2j/persist/pl/AssemblyInfo.cs has some figures/strings you just need to update. I am not aware at this moment if the C# allows the assembly code to extract these, so we could return them directly, right from C# native code (but it should be possible I guess).

Since the .NET assembly is just a wrapper with no logic of its own it would be better I think to retrieve the implementation version from the converted jar.

#39 - 01/19/2017 02:30 PM - Greg Shah

Ovidiu: As of revision 11158, is there anything else that is needed for C# support?

Hynek: The last known work for this is Eric's request from note 29.

#40 - 01/19/2017 02:43 PM - Hynek Cihlar

Greg Shah wrote:

> Ovidiu: As of revision 11158, is there anything else that is needed for C# support?
>
> Hynek: The last known work for this is Eric's request from note 29.

Yes, I have this implemented, when I succeed in running the introduced UDF I will commit the change.

#41 - 01/19/2017 02:50 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> Ovidiu: As of revision 11158, is there anything else that is needed for C# support?

I guess not. I will put it to the test tomorrow in the morning.

#42 - 01/19/2017 04:26 PM - Hynek Cihlar

Hynek Cihlar wrote:

> Yes, I have this implemented, when I succeed in running the introduced UDF I will commit the change.

I got stuck with installing Pl/Java for PostgreSQL. Following the official wiki documentation at Chapter_3_Development_Environment_Setup unfortunately leads to errors. I'll continue tomorrow.

#43 - 01/19/2017 04:36 PM - Greg Shah

I'm not sure if it will help, but you can look at #1866-6 where there are some more details about postgresql setup than we have yet covered in our documentation.

#44 - 01/19/2017 04:41 PM - Hynek Cihlar

Greg Shah wrote:

> I'm not sure if it will help, but you can look at #1866-6 where there are some more details about postgresql setup than we have yet covered in our documentation.

Greg, thanks for the link, the instructions seem to be more up to date.

#45 - 01/20/2017 07:39 AM - Ovidiu Maxiniuc

Ovidiu Maxiniuc wrote:

> Greg Shah wrote:
>
> > Ovidiu: As of revision 11158, is there anything else that is needed for C# support?
>
> I guess not. I will put it to the test tomorrow in the morning.

C# code is working fine. After installing it (it took me a little time to prepare the assemblies) I can do the following in the MS SQL Management Studio:

```
select  dbo.getFWDVersion_() as 'FWD Version'
    FWD Version
    FWD v3.0.0_p2j_2699a_11158
```

#46 - 01/20/2017 07:40 AM - Hynek Cihlar

Ovidiu Maxiniuc wrote:

> C# code is working fine. After installing it (it took me a little time to prepare the assemblies) I can do the following in the MS SQL Management Studio:
> [...]

Wonderful, thanks Ovidiu.

#47 - 01/20/2017 09:26 AM - Greg Shah

Are you having issues with the persistence startup change? Is there something we can do to help?

This is the last thing to include. I'd like to get 2699a rebased and into testing so that we can get it into the trunk.

#48 - 01/20/2017 09:52 AM - Hynek Cihlar

Greg Shah wrote:

> Are you having issues with the persistence startup change? Is there something we can do to help?

I have my environment properly set up, the new UDF function gets called correctly, but the implementation won't find the manifest with the version string. I am debugging the process now to see what's going on. I am tempted to have the version string generated to a static class field instead of the jar's manifest.

#49 - 01/20/2017 10:13 AM - Eric Faulhaber

Is this just a problem with PostgreSQL? Is it working with H2?

#50 - 01/20/2017 10:14 AM - Greg Shah

> I am tempted to have the version string generated to a static class field instead of the jar's manifest.

I'm fine with that. Actually, please do both so that we follow "the standard Java way" but we don't have to parse the manifest at runtime. This means that there will have to be a template for a Java class that gets reset on clean.

#51 - 01/20/2017 10:23 AM - Hynek Cihlar

Eric Faulhaber wrote:

> Is this just a problem with PostgreSQL? Is it working with H2?

The problem is PostgreSQL (PLJava) specific. When the UDF jar is installed with sqlj.install_jar(), the file is physically copied to the database. Since a non-standard class loader is used to read the jar from the DB and my implementation assumes a jar resource when searching for the correct manifest it is not resolved correctly.

One solution would be to enumerate all manifest resources and somehow identify the correct one (maybe with a special cookie value in the manifest itself). Another solution is to use the generated class field. I will go with the latter, while keeping the version in manifests, too, as Greg suggested.

#52 - 01/20/2017 02:11 PM - Constantin Asofiei

Hynek, please post what version text should I use for trunk P2J revision 11137 (to include it in the name of the zip files mentioned in https://proj.goldencode.com/issues/3209#note-555)

#53 - 01/20/2017 02:18 PM - Hynek Cihlar

Constantin Asofiei wrote:

> Hynek, please post what version text should I use for trunk P2J revision 11137 (to include it in the name of the zip files mentioned in https://proj.goldencode.com/issues/3209#note-555)

I am assuming you mean the p2j repository (i.e. not a customer specific repository), 3.0.0_p2j_trunk_11137.

#54 - 01/20/2017 04:11 PM - Hynek Cihlar

Eric Faulhaber wrote:

> Greg Shah wrote:
>
> > Eric: It seems to make sense that we would also read this p2jpl.jar value in server startup and output this value to the server log. This would also be a chance to detect if the p2jpl.jar was missing entirely and report that.
>
> Please add a package-private, static method to Persistence called reportUdfVersion. It should check out/in a database connection like we do in Persistence.queryIndexData. The SQL to get the version will just be select version() (or whatever your new version check UDF is called).

The UDF reporting is almost finished. One remaining issue is selecting the introduced UDF getFWDVersion.

Eric, the H2 UDF uses an overload name which is used to create the DB ALIAS. In my case it is getFWDVersion_1 and so JDBC SELECT getFWDVersion fails. I attempted to use a Hibernate Session to select the function, but that is still too low-level (see Persistence.reportUDFVersionConditionally() in 2699a revision 11159). I will need to somehow employ P2J's query processing to have the logical name correctly translated to the DB alias. Eric, please advise.

#55 - 01/22/2017 06:09 AM - Hynek Cihlar

Hynek Cihlar wrote:

> Eric, the H2 UDF uses an overload name which is used to create the DB ALIAS. In my case it is getFWDVersion_1 and so JDBC SELECT getFWDVersion fails. I attempted to use a Hibernate Session to select the function, but that is still too low-level (see Persistence.reportUDFVersionConditionally() in 2699a revision 11159). I will need to somehow employ P2J's query processing to have the logical name correctly translated to the DB alias. Eric, please advise.

I have committed a working solution to 2699a revision 11160 translating the HQL user-defined function name to SQL name with the newly added method HQLPreprocessor.getRegisteredFunction. I have put this version through regression testing.

Eric, please review.

#56 - 01/22/2017 10:08 PM - Eric Faulhaber

Sorry, didn't see your question in note 54 on Friday.

Code review 2699a/11160:

The code you added to HQLPreprocessor looks good, but I'd prefer you rework the changes to DatabaseManager and Persistence to not use Hibernate.

Did my suggested implementation of how the check out and in a temporary database connection in #28-29 not make sense or cause problems? I wanted you to do it this way in order to avoid using Hibernate and having a Persistence$Context instance created for the server startup thread. I think that instance and all the baggage associated with it, including the Hibernate session, will stay around for the life of the server. We don't want that just to get the version number.

#57 - 01/23/2017 02:29 AM - Hynek Cihlar

Eric Faulhaber wrote:

> Did my suggested implementation of how the check out and in a temporary database connection in #28-29 not make sense or cause problems? I wanted you to do it this way in order to avoid using Hibernate and having a Persistence$Context instance created for the server startup thread. I think that instance and all the baggage associated with it, including the Hibernate session, will stay around for the life of the server. We don't want that just to get the version number.

Due to the HQL->SQL method name translation I thought I would need Hibernate to properly call the UDF function. But then I found the HQLPreprocessor and the translation table. I will rework the query code to use plain JDBC.

#58 - 01/23/2017 03:33 AM - Hynek Cihlar

Hynek Cihlar wrote:

> Eric Faulhaber wrote:
>
> > Did my suggested implementation of how the check out and in a temporary database connection in #28-29 not make sense or cause problems? I wanted you to do it this way in order to avoid using Hibernate and having a Persistence$Context instance created for the server startup thread. I think that instance and all the baggage associated with it, including the Hibernate session, will stay around for the life of the server. We don't want that just to get the version number.
>
> Due to the HQL->SQL method name translation I thought I would need Hibernate to properly call the UDF function. But then I found the HQLPreprocessor and the translation table. I will rework the query code to use plain JDBC.

Eric, please review the changes in 2699a revision 11162. Also please note that I had to move the call to Persistence.reportUDFVersion() to the point where the UDF name aliases are already registered.

#59 - 01/23/2017 10:02 AM - Eric Faulhaber

Code review 2699a/11164:

Changes look good. One request: could you please add logging of the SQLException in Persistence.reportUDFVersion at debug level. We still want to report "undefined" version in this case, but this gives an option to log the actual cause by changing the logging level. No need to restart/rerun regression testing for this change.

#60 - 01/23/2017 02:21 PM - Hynek Cihlar

Eric Faulhaber wrote:

> Code review 2699a/11164:
>
> Changes look good. One request: could you please add logging of the SQLException in Persistence.reportUDFVersion at debug level. We still want to report "undefined" version in this case, but this gives an option to log the actual cause by changing the logging level. No need to restart/rerun regression testing for this change.

Logging added to 2699a revision 11165. Also the branch has passed regression tests.

#61 - 01/23/2017 04:26 PM - Hynek Cihlar

Task branch 2699a was merged to trunk as revision 11138 and archived.

#62 - 01/23/2017 04:26 PM - Hynek Cihlar

- % Done changed from 0 to 100

#63 - 01/23/2017 04:27 PM - Greg Shah

- Status changed from WIP to Closed