Database - Feature #4056

instrument FWD and supporting libraries to measure database performance

05/01/2019 02:11 AM - Eric Faulhaber

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Eric Faulhaber	**% Done:**	90%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 05/01/2019 02:21 AM - Eric Faulhaber

In order to improve the performance of the persistence layer of FWD, we first need to measure the time spent at several levels of the API, to identify bottlenecks. This can be accomplished through sampling, profiling (tracing), logging, and special purpose instrumentation, which may be temporary or permanent.

This task is about instrumenting the FWD persistence runtime and its supporting libraries to record such measurements.

Persistence performance should be measured at several key boundaries, such that we know the time being spent between each corresponding enter/return pair below:

- FWD persistence API (enter)
  - Hibernate API (enter)
    - JDBC API (enter)
      - socket/network (enter)
        - PostgreSQL server (enter)
          - parse/prepare work
          - execute work
        - PostgreSQL server (return)
      - socket/network (return)
    - JDBC API (return)
  - Hibernate API (return)
- FWD persistence API (return)

We have some existing logging in the Persistence class, but this needs to be updated to use high resolution timing, and to increase API coverage. I think currently only the query APIs are covered.

The facilities available at the Hibernate and the JDBC driver levels need to be explored, though I suspect we will be adding instrumentation there as well (probably temporarily).

Task branch 4056a has been created for this work.

#2 - 05/08/2019 12:51 PM - Eric Faulhaber

- Status changed from New to WIP

- Assignee set to Eric Faulhaber

Hynek, I am using AspectJ to capture method-level profile data (just number of invocations and aggregate elapsed time for now).

My prototype aspect defines execution pointcuts for all public methods by package (e.g., execution(public * com.goldencode.p2j.persist.*.*(..))). This works fine for FWD classes, but I understand that to make it work for third party classes (e.g., Hibernate, the JDBC driver, etc.) without recompiling them, I need to enable load-time weaving. Do you have any example of this already in FWD? Do you have any concern that this would introduce too much overhead for an aspect whose only purpose is to capture profile information?

#3 - 05/08/2019 01:07 PM - Hynek Cihlar

Eric Faulhaber wrote:

> Hynek, I am using AspectJ to capture method-level profile data (just number of invocations and aggregate elapsed time for now).
>
> My prototype aspect defines execution pointcuts for all public methods by package (e.g., execution(public * com.goldencode.p2j.persist.*.*(..))). This works fine for FWD classes, but I understand that to make it work for third party classes (e.g., Hibernate, the JDBC driver, etc.) without recompiling them, I need to enable load-time weaving. Do you have any example of this already in FWD? Do you have any concern that this would introduce too much overhead for an aspect whose only purpose is to capture profile information?

Yes, I use load-time weaving for running the uast test cases directly from my IDE. The reason was that I had some troubles with using the aspect compiler from the IDE. To enable load-time weaving (1) add -javaagent:/your-path-to/aspectjweaver.jar to your VM command line options. And (2) add META-INF/aop.xml to your class path (the simplest is to add it to the fwd.jar). With the aop.xml you define your aspects and include/exclude the target classes. The format is pretty straight forward, see my aop.xml below.

```
<aspectj>
    <aspects>
        <aspect name="com.goldencode.p2j.aspects.ui.SyncConfigChangesAspect"/>
        <aspect name="com.goldencode.p2j.aspects.ui.ConfigFieldSetterAspect"/>
        <aspect name="com.goldencode.p2j.aspects.ui.SyncCoordinatesAspect"/>
        <aspect name="com.goldencode.p2j.aspects.ui.UIStatementsAspect"/>
        <aspect name="com.goldencode.p2j.ui.client.widgetbrowser.WidgetBrowserAspect"/>
    </aspects>

<!-- this is useful when debugging the weaving process
    <weaver options="-verbose -showWeaveInfo">
        <include within="com.goldencode..*"/>
    </weaver>
-->

    <weaver>
        <include within="com.goldencode..*"/>
        <exclude within="*FieldAccess"/>
        <exclude within="*MethodAccess"/>
    </weaver>

</aspectj>
```

As of load-time performance, the FWD start time overhead for the aop.xml above is negligible. For runtime performance it is hard to say. Obviously it will depend on the complexity of the profiling logic as well as the call frequency of the monitored methods.

#4 - 05/08/2019 01:12 PM - Hynek Cihlar

Hynek Cihlar wrote:

> Eric Faulhaber wrote:
>
> > Hynek, I am using AspectJ to capture method-level profile data (just number of invocations and aggregate elapsed time for now).
> >
> > My prototype aspect defines execution pointcuts for all public methods by package (e.g., execution(public * com.goldencode.p2j.persist.*.*(..))). This works fine for FWD classes, but I understand that to make it work for third party classes (e.g., Hibernate, the JDBC driver, etc.) without recompiling them, I need to enable load-time weaving. Do you have any example of this already in FWD? Do you have any concern that this would introduce too much overhead for an aspect whose only purpose is to capture profile information?
>
> Yes, I use load-time weaving for running the uast test cases directly from my IDE. The reason was that I had some troubles with using the aspect compiler from the IDE. To enable load-time weaving (1) add -javaagent:/your-path-to/aspectjweaver.jar to your VM command line options. And (2) add META-INF/aop.xml to your class path (the simplest is to add it to the fwd.jar). With the aop.xml you define your aspects and include/exclude the target classes. The format is pretty straight forward, see my aop.xml below.
>
> [...]

I'm not sure how well you can combine load-time and compile-time weaving. I would say you should not include the aspects that have been weaved during compile. So if you run FWD from the standard fwd.jar, only include your new aspect(s) in the <aspect> element in aop.xml. Also you should leave the excludes in <weaver> XML element, these are for ReflectASM.

#5 - 05/08/2019 01:14 PM - Hynek Cihlar

Here is more info on load time weaving http://www.eclipse.org/aspectj/doc/released/devguide/ltw.html.

#6 - 05/08/2019 01:32 PM - Eric Faulhaber

Thank you for the guidance.

I am using compile-time weaving only because that seemed to be the most direct way to get started understanding the framework. Also, it seemed intuitive that the runtime overhead using compile-time weaving would be lower than load-time weaving. Do I need to fundamentally change my aspect implementation to use only load-time weaving? Currently, it consists of several (empty) pointcut method definitions to specify the methods I want to profile (by package) and an @Around advice which does the data collection work. The target packages will not change often, if at all, and it is ok to recompile FWD to update them (see below).

I should have mentioned: it is necessary to have zero overhead for production/normal use, so the intention is to have this infrastructure entirely disabled by default and only built into FWD when a particular ant/gradle target is used to build. I haven't worked this part out yet, but is anything in the

approach you outline above likely to make the "disabled by default" goal harder to achieve? I hadn't initially considered the XML file approach, given there is no deploy-time configuration need. I had been looking at the custom classloader approach initially, but I hadn't gotten too far down that road before I realized I needed some advice. I'm quite new to the framework, so I hadn't settled on any particular approach yet.

#7 - 05/08/2019 01:40 PM - Hynek Cihlar

Eric Faulhaber wrote:

> Thank you for the guidance.
>
> I am using compile-time weaving only because that seemed to be the most direct way to get started understanding the framework. Also, it seemed intuitive that the runtime overhead using compile-time weaving would be lower than load-time weaving. Do I need to fundamentally change my aspect implementation to use only load-time weaving?

I don't think the aspect implementation should change in any way for LTW.

> I should have mentioned: it is necessary to have zero overhead for production/normal use, so the intention is to have this infrastructure entirely disabled by default and only built into FWD when a particular ant/gradle target is used to build. I haven't worked this part out yet, but is anything in the approach you outline above likely to make the "disabled by default" goal harder to achieve? I hadn't initially considered the XML file approach, given there is no deploy-time configuration need. I had been looking at the custom classloader approach initially, but I hadn't gotten too far down that road before I realized I needed some advice. I'm quite new to the framework, so I hadn't settled on any particular approach yet.

I don't know if there is any advantage of using the custom class loader approach over the aop.xml file. The zero-overhead could be achieved during build by either including the aop.xml in the target jar or not.

#8 - 05/08/2019 01:56 PM - Eric Faulhaber

Hynek Cihlar wrote:

> I'm not sure how well you can combine load-time and compile-time weaving.

Actually, I did nothing explicitly to enable compile-time weaving. The profile data collection just worked (for FWD classes only, not third party classes) when I added my aspect class to the com.goldencode.p2j.aspects package, presumably because you previously set it up.

> I would say you should not include the aspects that have been weaved during compile.

I only have one aspect currently, which is intended to do the same thing whether a method is in a FWD class or in a third party class. How do I

exclude this from compile-time weaving, so I can use it with load-time weaving? As noted above, it already seems to be part of the compile-time weaving you've enabled previously.

> So if you run FWD from the standard fwd.jar, only include your new aspect(s) in the <aspect> element in aop.xml. Also you should leave the excludes in <weaver> XML element, these are for ReflectASM.

Ok.

#9 - 05/08/2019 02:01 PM - Hynek Cihlar

Eric Faulhaber wrote:

> Hynek Cihlar wrote:
>
> > I'm not sure how well you can combine load-time and compile-time weaving.
>
> Actually, I did nothing explicitly to enable compile-time weaving. The profile data collection just worked (for FWD classes only, not third party classes) when I added my aspect class to the com.goldencode.p2j.aspects package, presumably because you previously set it up.
>
> > I would say you should not include the aspects that have been weaved during compile.
>
> I only have one aspect currently, which is intended to do the same thing whether a method is in a FWD class or in a third party class. How do I exclude this from compile-time weaving, so I can use it with load-time weaving? As noted above, it already seems to be part of the compile-time weaving you've enabled previously.

How about to exclude the FWD classes in aop.xml? So that the FWD classes are only weaved by the compiler. Also I found this in the documentation, "As of AspectJ 5 aspects (code style or annotation style) and woven classes are reweavable by default.".

#10 - 05/08/2019 02:30 PM - Eric Faulhaber

Hynek Cihlar wrote:

> How about to exclude the FWD classes in aop.xml? So that the FWD classes are only weaved by the compiler.

This makes sense for this purpose, but is it potentially limiting for future use? I mean, the excludes seem to apply to all aspects (current and future) included in the <aspects> element. But, I guess it would make sense to compile-time weave the FWD classes for anything for which we would write new aspects in the future, for the same reason we are doing it now. In this case, I still need to exclude this aspect from the default FWD build, since excluding the aop.xml file from the jar would not be enough to disable it.

I assume your example XML above is illustrative, and I do not need to include the existing UI aspects in the <aspects> element, since they are compile-time weaved. Correct?

So, my XML would look something like this:

```
<aspectj>
    <aspects>
        <aspect name="com.goldencode.p2j.aspects.MethodProfileAspect"/>
    </aspects>

<!-- this is useful when debugging the weaving process
    <weaver options="-verbose -showWeaveInfo">
        <include within="com.goldencode..*"/>
    </weaver>
-->

    <weaver>
        <exclude within="com.goldencode..*"/>
        <exclude within="*FieldAccess"/>
        <exclude within="*MethodAccess"/>
    </weaver>

</aspectj>
```

Is that what you meant?

> Also I found this in the documentation, "As of AspectJ 5 aspects (code style or annotation style) and woven classes are reweavable by default.".

I noticed that, but I wasn't sure what it meant. I think I understand now.

#11 - 05/08/2019 02:51 PM - Hynek Cihlar

Eric Faulhaber wrote:

> Hynek Cihlar wrote:
>
> > How about to exclude the FWD classes in aop.xml? So that the FWD classes are only weaved by the compiler.
>
> This makes sense for this purpose, but is it potentially limiting for future use? I mean, the excludes seem to apply to all aspects (current and future) included in the <aspects> element. But, I guess it would make sense to compile-time weave the FWD classes for anything for which we would write new aspects in the future, for the same reason we are doing it now. In this case, I still need to exclude this aspect from the default FWD build, since excluding the aop.xml file from the jar would not be enough to disable it.

True, the aspect itself should also be removed from the build to have the "disabled-by-default" logic. We do the same with WidgetBrowserAspect class, see widget.browser.exclude-path in build.xml.

> I assume your example XML above is illustrative, and I do not need to include the existing UI aspects in the <aspects> element, since they are compile-time weaved. Correct?

Correct.

> So, my XML would look something like this:
>
> [...]
>
> Is that what you meant?

Yes. You may also need to add the include XML elements to narrow the search for the classes to be woven and improve load times.

#12 - 06/10/2019 02:12 AM - Eric Faulhaber

- File trace.csv added

Attached is some Hotel GUI sample trace output showing aggregate times (nanosec) and invocation counts by top-level traced method, bucketed by FWD persistence layer, Hibernate, c3p0, and H2. PostgreSQL JDBC driver output is not included, since this deployment used H2 as the back end. However, the PostgreSQL JDBC driver would have been traced, had the back end been PostgreSQL.

The code is checked into branch 4056a as revision 11308, but it needs cleanup, some doc, and a rebase. I'd also like to fix the sorting, so we see aggregate times in descending order by default. The CSV output is done inline from the aspect class using some hard-coded values. Although this is a nasty/lazy design, I don't intend to address it at this time. There are some errors in the server log for methods which AspectJ was not able to load-time weave. I don't know the exact cause of this, but I don't think it's a big problem in the overall scheme of the feature set, so I'm not planning on addressing it at this point.

I've made a companion commit to Hotel GUI (rev 199) to provide an example of how to integrate this feature set into a conversion project. The changes are additive, so this commit can still be used without FWD branch 4056a. The new additions are a -a option to server.sh (run with AOP tracing available but disabled by default); and -m1 and -m0 to turn active tracing on and off, respectively. The latter two options must be run from a separate command line and will remotely attach to the server running under the same instance number. It should be noted that if the server is not run with the -a option, the tracing feature is completely disabled, so there is no added overhead for normal use. We use AspectJ load-time weaving when -a is provided; otherwise, the tracing aspect is not loaded at all.

The aggregate times of each bucket include those of the buckets "beneath" them (i.e., methods in other buckets called by them). Although the bucket columns in the CSV file are arranged from left to right in the order they would most often be called, note that a call-back initiated by a "lower-level" bucket method could invoke a method in a "higher-level" bucket, leading to aggregate times which may seem counter-intuitive. For example, if a Hibernate method invoked a registered FWD method via a callback hook, the Hibernate method would show a larger aggregate time than the FWD method. I don't know of any such cases off the top of my head, but it is theoretically possible.

#13 - 06/10/2019 09:12 AM - Greg Shah

This looks very useful. Some suggested improvements for the output:

- Instead of Agg Time use Agg Time (nanosec) so that the reader doesn't have to know the scale.
- Add a % of Total column for each of the "beneath" buckets. This allows the reader to avoid some mental math to get the sense of the contribution to the total Persistence number.
- Add a Totals row that accumulates/calculates the multiple Agg Time (nanosec), Count and % of Total columns.
- Add a row at the bottom that reports the total time that tracing was active (the nanoseconds between -m1 and -m0@). When the user matches testing with "quick" enable/disable of tracing, then this number will be very close to the total amount of time for the test scenario. Having that included with the tracing results is better than trying to remember the context separately. In cases where it doesn't match, you aren't losing anything. The user can always edit the output to save the real number. But in cases where it is close, then this allows future review of the results to be more effective.

#14 - 06/12/2019 03:35 PM - Eric Faulhaber

I've added all the suggested features except the Totals row (will add in a separate pass). I've rebased 4056a to trunk rev 11315. Current branch revision is 11318.

#15 - 06/14/2019 03:58 PM - Eric Faulhaber

Hynek, AspectJ is executing a join point for methods outside the scope I am trying to define. For instance, this dynamic proxy method:

```
public final void com.sun.proxy.$Proxy13.invalidate()
```

My aop.xml looks like this:

```
<aspectj>

   <aspects>
      <!-- a concrete aspect to define the abstract MethodTraceAspect.scope() pointcut -->
      <concrete-aspect name="com.goldencode.p2j.aspects.ltw.__MethodTraceAspect"
                       extends="com.goldencode.p2j.aspects.ltw.MethodTraceAspect">
         <pointcut name="scope"
                   expression="execution(* com.goldencode.p2j.persist..*.*(..))
                            || execution(* org.hibernate..*.*(..))
                            || execution(* org.postgresql..*.*(..))
                            || execution(* org.h2..*.*(..))
                            || execution(* com.mchange..*.*(..))"
         />
      </concrete-aspect>
   </aspects>

<!-- this is useful when debugging the weaving process>
   <weaver options="-verbose -showWeaveInfo"/>
-->

   <weaver>
      <exclude within="com.goldencode.p2j.aspects.ui..*"/>
      <exclude within="com.goldencode.p2j.ui..*"/>
      <exclude within="*FieldAccess"/>
      <exclude within="*MethodAccess"/>
      <exclude within="com.goldencode.p2j.persist.TableWrapper"/>
      <exclude within="com.goldencode.p2j.persist.PropertyDefinition"/>
   </weaver>

</aspectj>
```

Based on it hitting the public final void com.sun.proxy.$Proxy13.invalidate() pointcut above, and the fact that I had to explicitly exclude com.goldencode.p2j.aspects.ui..* and com.goldencode.p2j.ui..*, AspectJ seems to be load-time weaving more than I want. I'm just trying to run my aspect against the methods in:

```
com.goldencode.p2j.persist.*
org.hibernate.*
org.postgresql.*
org.h2.*
com.mchange.*
```

These do seem to be woven, but it seems to be doing much more than this scope. Any idea why it's load-time-weaving more than this, based on my configuration above?

com.goldencode.p2j.aspects.ltw.MethodTraceAspect is an abstract class, with an abstract pointcut defined as follows:

```
   @Pointcut
   public abstract void scope();
```

The abstract class/aspect is overridden by the concrete aspect com.goldencode.p2j.aspects.ltw.__MethodTraceAspect, defined in aop.xml above.

#16 - 06/14/2019 04:20 PM - Hynek Cihlar

Eric Faulhaber wrote:

> Hynek, AspectJ is executing a join point for methods outside the scope I am trying to define. For instance, this dynamic proxy method:

Eric, did you try to enable the -verbose -showWeaveInfo weaver options in aop.xml? Maybe it will give some clues.

#17 - 06/14/2019 11:16 PM - Eric Faulhaber

It told me that the com.sun.proxy.* join point weaves were ... advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java). However, I still don't really understand why. Anyway, I added an exclude directive explicitly for these and it no longer weaves them.

#18 - 07/07/2019 04:44 PM - Eric Faulhaber

Task branch 4056a has been rebased to trunk rev 11322.

#19 - 07/12/2019 02:24 AM - Eric Faulhaber

- % Done changed from 0 to 90

Branch 4056a was merged to trunk revision 11323. It passed ChUI conversion and runtime regression testing (the latter was run after the merge). The branch has been archived.

TODO: in a new branch, move MethodTraceController out of the aspects.ltw package and from fwdaopltw.jar into p2j.jar. ATM, it creates a dependency on fwdaopltw.jar, which should only be needed if you are using method tracing (not the common case).

#20 - 10/19/2022 04:09 AM - Alexandru Lungu

I could set-up this feature with 3821c/rev.14305 and Hotel GUI. I am starting a standard server and then I attach the method tracing using server.sh -a -m1. The good part is that I get the following message in the standard server:

```
Method profiling is enabled; this feature is not intended for production use!
```

The "attaching server" exists with no error (I hope this is the expected behavior). However, it states that only some methods were weaved:

```
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(int com.goldencode.p2j.persist.Database.hashC
ode())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:279) advised by around advice from 'com.g
oldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.Database.e
quals(java.lang.Object))' in Type 'com.goldencode.p2j.persist.Database' (Database.java:310) advised by around
 advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
atabase.toString())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:355) advised by around advic
e from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
atabase.getName())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:383) advised by around advice
 from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
atabase.getId())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:393) advised by around advice f
rom 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.net.InetSocketAddress com.goldencode.p2j
.persist.Database.getSocketAddress())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:405) advis
```

ed by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.Database.isLocal())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:420) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.Database$Type com.goldencode.p2j.persist.Database.getType())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:435) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.Database.isPrimary())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:445) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.Database.isDirty())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:455) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.Database.isMeta())' in Type 'com.goldencode.p2j.persist.Database' (Database.java:465) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.Database.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.Database' (Database.java:480) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.Database.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.Database' (Database.java:501) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.Database$Type[] com.goldencode.p2j.persist.Database$Type.values())' in Type 'com.goldencode.p2j.persist.Database$Type' (Database.java:105) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.Database$Type com.goldencode.p2j.persist.Database$Type.valueOf(java.lang.String))' in Type 'com.goldencode.p2j.persist.Database$Type' (Database.java:105) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(int com.goldencode.p2j.persist.Database$Type.getId())' in Type 'com.goldencode.p2j.persist.Database$Type' (Database.java:133) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(long com.goldencode.p2j.persist.DatabaseConfig.getDeactivateIfNotUsedSec())' in Type 'com.goldencode.p2j.persist.DatabaseConfig' (DatabaseConfig.java:146) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatabaseConfig.isUseJavaUDFs())' in Type 'com.goldencode.p2j.persist.DatabaseConfig' (DatabaseConfig.java:155) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.orm.Settings com.goldencode.p2j.persist.DatabaseConfig.getOrmSettings())' in Type 'com.goldencode.p2j.persist.DatabaseConfig' (DatabaseConfig.java:165) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DatabaseConfig.getSchema())' in Type 'com.goldencode.p2j.persist.DatabaseConfig' (DatabaseConfig.java:175) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatabaseConfig.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.DatabaseConfig' (DatabaseConfig.java:190) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatabaseConfig.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.DatabaseConfig' (DatabaseConfig.java:209) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatasetWrapper.setUseBeforeImage(boolean))' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:192) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatasetWrapper.isUseBeforeImage())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:202) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatasetWrapper.setXmlDataset(java.lang.String))' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:213) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DatasetWrapper.getXmlDataset())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:223) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatasetWrapper.isAsXml())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:233) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatasetWrapper.setAsXml(boolean))' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:244) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.DataSetContainer com.goldencode.p2j.persist.DatasetWrapper.getResource())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (

DatasetWrapper.java:254) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (M
ethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.Dat
asetWrapper.getTableDefs())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:264) adv
ised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.Dat
asetWrapper.getRelationsDefs())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:275)
 advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatasetWra
pper.isInput())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:287) advised by arou
nd advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatasetWra
pper.isOutput())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:305) advised by aro
und advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatasetWra
pper.isAppend())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:322) advised by aro
und advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatasetWra
pper.isDatasetHandle())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:338) advised
 by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DatasetWra
pper.isFromJson())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:348) advised by a
round advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatasetWrappe
r.setDatasetHandle(boolean))' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:359) ad
vised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
atasetWrapper.getDatasetName())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:369)
 advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatasetWrappe
r.setDatasetName(java.lang.String))' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:
380) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.jav
a)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatasetWrappe
r.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.ja
va:395) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.
java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DatasetWrappe
r.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java
:482) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.ja
va)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.TableWrapper com.g
oldencode.p2j.persist.DatasetWrapper.getTableWrapper(int))' in Type 'com.goldencode.p2j.persist.DatasetWrapper
' (DatasetWrapper.java:537) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect'
 (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
atasetWrapper.toString())' in Type 'com.goldencode.p2j.persist.DatasetWrapper' (DatasetWrapper.java:597) advis
ed by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TableResultSe
t.setRows(java.util.List))' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:235) advi
sed by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TableResultSe
t.setRowsMeta(java.util.List))' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:246)
advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.Tab
leResultSet.getRows())' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:257) advised
by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.Tab
leResultSet.getRowsMeta())' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:268) advi
sed by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TableResultSe
t.setProperties(java.util.List))' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:279
) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.Tab
leResultSet.getProperties())' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:290) ad
vised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.sql.ResultSet com.goldencode.p2j.persist
.TableResultSet.asResultSet())' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:301)
advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TableResultSe
t.clearRows())' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:320) advised by aroun
d advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.Tab
leResultSet.access$000(com.goldencode.p2j.persist.TableResultSet))' in Type 'com.goldencode.p2j.persist.TableR
esultSet' (TableResultSet.java:113) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTrac

eAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.TableResultSet.access$100(com.goldencode.p2j.persist.TableResultSet))' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:113) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.logging.Logger com.goldencode.p2j.persist.TableResultSet.access$200())' in Type 'com.goldencode.p2j.persist.TableResultSet' (TableResultSet.java:113) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.MassiveContainer.setStructureName(java.lang.String))' in Type 'com.goldencode.p2j.persist.MassiveContainer' (MassiveContainer.java:127) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.MassiveContainer.getStructureName())' in Type 'com.goldencode.p2j.persist.MassiveContainer' (MassiveContainer.java:137) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.MassiveContainer.isInput())' in Type 'com.goldencode.p2j.persist.MassiveContainer' (MassiveContainer.java:147) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.MassiveContainer.isOutput())' in Type 'com.goldencode.p2j.persist.MassiveContainer' (MassiveContainer.java:157) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.MassiveContainer.isAppend())' in Type 'com.goldencode.p2j.persist.MassiveContainer' (MassiveContainer.java:167) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TempTableResultSet.setRows(java.util.List))' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:237) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TempTableResultSet.setRowsMeta(java.util.List))' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:251) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TempTableResultSet.setProperties(java.util.List))' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:265) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.TempTableResultSet.hasMoreProperties())' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:280) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.PropertyDefinition com.goldencode.p2j.persist.TempTableResultSet.nextProperty())' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:293) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.TempTableResultSet.hasMoreRows())' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:306) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.Object[] com.goldencode.p2j.persist.TempTableResultSet.nextRow())' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:319) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.Object[] com.goldencode.p2j.persist.TempTableResultSet.nextRowMeta())' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:337) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TempTableResultSet.clearRows())' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:346) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TempTableResultSet.init(boolean))' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:367) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TempTableResultSet.lambda$init$0(java.util.List, java.util.List, java.lang.Object[][]))' in Type 'com.goldencode.p2j.persist.TempTableResultSet' (TempTableResultSet.java:437) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DsRelationDefinition.getName())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:151) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setName(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:162) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)

[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DsRelationDefinition.getParentBuffer())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:172) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setParentBuffer(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:183) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DsRelationDefinition.getChildBuffer())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:193) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setChildBuffer(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:204) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DsRelationDefinition.getWhereString())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:214) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setWhereString(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:225) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setFieldPairs(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:236) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setFkeyHidden(boolean))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:247) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setRecursive(boolean))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:258) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setNested(boolean))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:269) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setReposition(boolean))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:280) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsRelationDefinition.isActive())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:290) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDefinition.setActive(boolean))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:301) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsRelationDefinition.isParentId())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:311) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DsRelationDefinition.getFieldPairs())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:321) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsRelationDefinition.isFkeyHidden())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:331) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsRelationDefinition.isRecursive())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:341) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsRelationDefinition.isNested())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:351) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsRelationDefinition.isReposition())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition.java:361) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.

```
java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDef
inition.setParentId(boolean))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefinition
.java:372) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspe
ct.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDef
inition.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRel
ationDefinition.java:387) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (
MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsRelationDef
inition.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelat
ionDefinition.java:412) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (Me
thodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
sRelationDefinition.toString())' in Type 'com.goldencode.p2j.persist.DsRelationDefinition' (DsRelationDefiniti
on.java:433) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAs
pect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.DsTableDefinition
com.goldencode.p2j.persist.DsTableDefinition.getPeerDef())' in Type 'com.goldencode.p2j.persist.DsTableDefinit
ion' (DsTableDefinition.java:237) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceA
spect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefini
tion.setPeerDef(com.goldencode.p2j.persist.DsTableDefinition))' in Type 'com.goldencode.p2j.persist.DsTableDef
inition' (DsTableDefinition.java:254) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTr
aceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
sTableDefinition.getName())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:26
6) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefini
tion.setName(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.jav
a:277) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.j
ava)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsTableDef
inition.isAfterTable())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:287) a
dvised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsTableDef
inition.isBeforeTable())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:297)
advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.DsTableDef
inition.isSimpleTable())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:307)
advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(int com.goldencode.p2j.persist.DsTableDefinit
ion.getNumFields())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:323) advis
ed by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(int com.goldencode.p2j.persist.DsTableDefinit
ion.getNumIndexes())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:333) advi
sed by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
sTableDefinition.getXmlns())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:3
43) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java
)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
sTableDefinition.getXmlPrefix())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.ja
va:353) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.
java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
sTableDefinition.getXmlNodeName())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.
java:363) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspec
t.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
sTableDefinition.getErrorString())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.
java:373) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspec
t.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.D
sTableDefinition.getIndexes())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java
:385) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.ja
va)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefini
tion.setXmlns(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.ja
va:396) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.
java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefini
tion.setXmlPrefix(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinitio
n.java:407) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAsp
ect.java)
```

[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefinition.setXmlNodeName(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:418) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefinition.setErrorString(java.lang.String))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:429) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefinition.setIndexes(int, java.lang.String))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:442) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.Iterator com.goldencode.p2j.persist.DsTableDefinition.propertyIterator())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:455) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.Iterator com.goldencode.p2j.persist.DsTableDefinition.rowIterator())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:477) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.DsTableDefinition.getProperties())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:499) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.DsTableDefinition.getRows())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:516) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.List com.goldencode.p2j.persist.DsTableDefinition.getRowsMeta())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:534) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.TableResultSet com.goldencode.p2j.persist.DsTableDefinition.getResultSet())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:550) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefinition.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:565) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefinition.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:650) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.DsTableDefinition.clearRows())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:722) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.DsTableDefinition.toString())' in Type 'com.goldencode.p2j.persist.DsTableDefinition' (DsTableDefinition.java:739) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.util.logical com.goldencode.p2j.persist.DataSourceModifiable.isDataSourceModified(com.goldencode.p2j.persist.Buffer))' in Type 'com.goldencode.p2j.persist.DataSourceModifiable' (DataSourceModifiable.java:102) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.orm.IndexState.get(int))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:119) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.IndexState.set(int, boolean))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:132) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.IndexState.clear(int))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:143) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.orm.IndexState.isEmpty())' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:154) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.orm.IndexState.getPrimary())' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:164) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.IndexState.setAll(int, int))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:177) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.IndexState.clearAll())' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:193) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)

[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.BitSet com.goldencode.p2j.persist.orm.IndexState.getBitSet(boolean))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:208) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.BitSet com.goldencode.p2j.persist.orm.IndexState.bitSetFromUid(int))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:222) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(int com.goldencode.p2j.persist.orm.IndexState.offsetFromUid(int))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:235) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.IndexState.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:250) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.IndexState.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.orm.IndexState' (IndexState.java:269) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.io.Serializable com.goldencode.p2j.persist.RecordIdentifier.getTable())' in Type 'com.goldencode.p2j.persist.RecordIdentifier' (RecordIdentifier.java:145) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.Long com.goldencode.p2j.persist.RecordIdentifier.getRecordID())' in Type 'com.goldencode.p2j.persist.RecordIdentifier' (RecordIdentifier.java:155) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.RecordIdentifier.equals(java.lang.Object))' in Type 'com.goldencode.p2j.persist.RecordIdentifier' (RecordIdentifier.java:170) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(int com.goldencode.p2j.persist.RecordIdentifier.hashCode())' in Type 'com.goldencode.p2j.persist.RecordIdentifier' (RecordIdentifier.java:193) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.RecordIdentifier.toString())' in Type 'com.goldencode.p2j.persist.RecordIdentifier' (RecordIdentifier.java:204) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.RecordIdentifier.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.RecordIdentifier' (RecordIdentifier.java:219) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.RecordIdentifier.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.RecordIdentifier' (RecordIdentifier.java:238) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(com.goldencode.p2j.persist.orm.Settings com.goldencode.p2j.persist.orm.Settings.get(com.goldencode.p2j.persist.Database))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:143) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.String com.goldencode.p2j.persist.orm.Settings.getString(java.lang.String, java.lang.String))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:224) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.Integer com.goldencode.p2j.persist.orm.Settings.getInteger(java.lang.String, java.lang.Integer))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:256) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.lang.Boolean com.goldencode.p2j.persist.orm.Settings.getBoolean(java.lang.String, java.lang.Boolean))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:281) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.orm.Settings.put(java.lang.String, java.lang.Object))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:299) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(java.util.Map com.goldencode.p2j.persist.orm.Settings.getAllForCategory(java.lang.String))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:314) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.Settings.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:329) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.orm.Settings.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.orm.Settings' (Settings.java:346) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(boolean com.goldencode.p2j.persist.TableSignature.equals(java.lang.Object))' in Type 'com.goldencode.p2j.persist.TableSignature' (TableSignature.java:155) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.java)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TableSignatur

```
e.readExternal(java.io.ObjectInput))' in Type 'com.goldencode.p2j.persist.TableSignature' (TableSignature.java
:184) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.ja
va)
[AppClassLoader@18b4aac2] weaveinfo Join point 'method-execution(void com.goldencode.p2j.persist.TableSignatur
e.writeExternal(java.io.ObjectOutput))' in Type 'com.goldencode.p2j.persist.TableSignature' (TableSignature.ja
va:202) advised by around advice from 'com.goldencode.p2j.aspects.ltw.__MethodTraceAspect' (MethodTraceAspect.
java)
```

My aop.xml is the same as in #4056-15, meant to weave eveything from persist and even more. However, the list shows only some methods from some classes in this package. For example, the AdaptiveQuery, PreselectQuery or Compoundquery are **not** in the list. For this matter, when disabling the tracing (server.sh -a -m0), a csv is genertated but without any entry. I also tried the -debug option of the weaver. The queries (adaptive, preselect or compound) are not considered for "debug weaving" nor for "debug not weaving" (basically they are not visible?):

- "A class loader may only weave classes that it defines. It may not weave classes loaded by a delegate or parent class loader." is a statement from the aspectj documentation of LTW. I am not sure how our com.goldencode.p2j.classloader.MultiClassLoader is interferring with this statement, but it doesn't really make sense to weave persist.Database and omit persist.AdaptiveQuery for instance.
- The last debug line from the weaving process is [AppClassLoader@18b4aac2] debug not weaving 'com.sun.proxy.$Proxy2'. Maybe this attempt silently halts the weaving process such that the other persist classes are not weaved.

Any hints into this matter?

#21 - 10/19/2022 04:01 PM - Eric Faulhaber

- File trace.csv added

TBH, I've never fully understood the AspectJ weaving implementation, though back when I initially implemented this, I recall it was weaving much more than the list you documented above. Also, it was generating CSVs containing considerable data. While this tracing code has been frozen in time since then, we have, of course, made many other changes in the meantime, elsewhere in the runtime. I don't think we've changed much with the class loading since then, however.

Did you try uncommenting the "this is useful when debugging the weaving process" line in aop.xml? Also note the same aop.xml is stored in fwdaopltw.jar, so you may need to remove this temporarily while working the kinks out of the process. At minimum, we want to remove the references to Hibernate and instead trace the time spent in com.goldencode.p2j.persist.orm and its sub-packages. I don't know how well the current, recursive implementation will work with that, considering it itself is a sub-package of com.goldencode.p2j.persist, which we also want to trace. But maybe it's enough to be aware of that organization when analyzing the data.

Finally, please note the pointcuts in aop.xml are duplicated in the com.goldencode.p2j.aspects.ltw.MethodTraceAspect class as the array of "bucket" keys to use when organizing the trace data. For now, unless we have a better way of extracting the list of packages we want to trace from the aop.xml configuration, we need to keep that bucket list in sync with the pointcuts in aop.xml.

For reference, I am attaching another trace.csv file that was generated after implementing most of the improvements Greg suggested in #4056-13. This one also was created using the Hotel GUI application, though I don't remember the activity within the application while tracing was enabled.

#22 - 10/20/2022 12:24 AM - Eric Faulhaber

Eric Faulhaber wrote:

> For reference, I am attaching another trace.csv file that was generated after implementing most of the improvements Greg suggested in #4056-13. This one also was created using the Hotel GUI application, though I don't remember the activity within the application while tracing was enabled.

The top priority is getting the trace data collection working again, but beyond that, if you have ideas about a better way to organize the trace data for analysis, I am open to them. The current format is difficult to review. At a high level, we are looking for an understanding of how the time spent in the persistence layer is allocated among the packages we are specifying, and the current output does that. But I am also interested in ways we could present the detailed data better.

#23 - 10/20/2022 05:17 AM - Greg Shah

> The queries (adaptive, preselect or compound) are not considered for "debug weaving" nor for "debug not weaving" (basically they are not visible?):
> ...
> Any hints into this matter?

I think this was by design. The idea of this particular kind of tracing was that we must only detect entrance to the persistence layer from outside of that layer. We don't want to capture data from internal calls within the persistence layer. This was because we were trying to measure the total time spent in persistence and below (at the time, Hibernate, PostgrSQL and H2). Method level tracing is very expensive and distorts the results, so we wanted to minimize the impact. I recall there was a lot of work involved to identify external calls from internal calls at runtime.

Eric: Am I mis-remembering?

#24 - 10/20/2022 06:23 AM - Alexandru Lungu

I got it now. It was a mistake from my part (I was not using -a option on the target server to be traced, but on the server run with -m1). As a short documentation on how to get this feature up for Hotel GUI:

- run server.sh to normally start an server without any kind of tracing
- run server.sh -a to start a server which is weaved and ready for method tracing
- run server.sh -m1 to enable the method tracing on the server started with -a
- run server.sh -m0 to disable the method tracing on the server started with -a. This also generates a csv with the results.

You are right, there seems to be more methods weaved than expected. Even if this is not a sound solution, at least it is complete.
Note that I am also capturing AdaptiveQuery for instance, but only AdaptiveQuery.initialize. This confirms the fact that only the entry points in the

"persistence" package are traced (maybe this is motivation for all the logic inside @MethodTraceAspect indeed).

#25 - 10/21/2022 09:39 AM - Dănuț Filimon

Committed 3821c/rev.14313. Added AspectJ on com.goldencode.p2j.persist.orm subpackage while keeping com.goldencode.p2j.persist..

I removed the reference to Hibernate from aop.xml and made a change to MethodTraceAspect.matchBucket. Instead of returning the **first** bucket that it finds, it will return the **best** match (if any) from all the existent buckets. It is important to mention that, if we want to test a subpackage, **it needs to placed under the parent package** in the buckets array (or lower). This doesn't cause any problems, but it provides a more understandable layout when writing the **trace.csv** file.

I will update Profiling wiki with a setup for Method Tracing soon.

#26 - 10/25/2022 02:10 AM - Eric Faulhaber

Dănuț Filimon wrote:

> Committed 3821c/rev.14313. Added AspectJ on com.goldencode.p2j.persist.orm subpackage while keeping com.goldencode.p2j.persist..

Please commit the same update to branch 6129b today. Thank you.

In the profiling wiki, there is a section stubbed out for an HTML UI. What does that entail? Has implementation of that begun? What is the estimated effort? I don't want to try to re-invent the wheel when it comes to CPU tracing, since there are products that do that already and do it well. But if we can make the data collected by this implementation a bit easier to view/interpret with a modest effort, that would be a welcome improvement.

#27 - 10/25/2022 04:03 AM - Alexandru Lungu

Eric Faulhaber wrote:

> Please commit the same update to branch 6129b today. Thank you.

Done it in 6129b/rev. 14306.

> In the profiling wiki, there is a section stubbed out for an HTML UI. What does that entail?

It took a bit of time to understand the layout of the results in the CSV. I am working on an exportToHTML method in the aspect to split the tracing per scenario/bucket in separate files (in a trace folder). Based on this structure, these can be displayed through a simple Web page.

> Has implementation of that begun?

Yes. The exportToHTML and most of the HTML work is ready. I am working on:

- Integrate per bucket totals
- Cutting out scenarios which have an own time < 1ms.
- Computing an overall time per scenario (the sum of all buckets per FWD API interaction)
- Allowing the client-based sorting of trace methods

- Edited: Add something like org.h2. (5) to symbolize that 5 methods were traced in the org.h2. package

  What is the estimated effort? I don't want to try to re-invent the wheel when it comes to CPU tracing, since there are products that do that already and do it well. But if we can make the data collected by this implementation a bit easier to view/interpret with a modest effort, that would be a welcome improvement.

I feel like they are more than a "bit easier" to interpret right now. I am keen on finishing this by the end of this day.

#28 - 10/25/2022 10:16 AM - Eric Faulhaber

Alexandru Lungu wrote:

> Eric Faulhaber wrote:
>
> > Please commit the same update to branch 6129b today. Thank you.
>
> Done it in 6129b/rev. 14306.

Thank you.

> I feel like they are more than a "bit easier" to interpret right now. I am keen on finishing this by the end of this day.

Great! I look forward to trying it.

#29 - 10/25/2022 11:53 AM - Alexandru Lungu

Committed 6129b/rev. 14307 including the new support for the HTML presentation of the method tracing data. There are still bits to be added, but the interface is now fully functional.
It was a bit challenging to provide this solution without requiring a file server (so I had to rework the approach). The modern browsers' CORS policy didn't allow the dynamic loading of data through import or fetch from JS. Therefore, most of the JS is included statically in the index.html. Also, all of the files are generated from the Java code (JS, CSS, HTML). Once the method tracing is complete, enter deploy/server/trace/index.html; it should work out-of-the-box (also ensure JS is enabled for the page). Including the documentation now in the wiki section.

#30 - 10/27/2022 01:03 AM - Eric Faulhaber

Alexandru, I like the HTML UI, nice work! Is the Grand Totals by package breakdown somewhere in there?

For a large customer application, here is the package breakdown from the CSV:

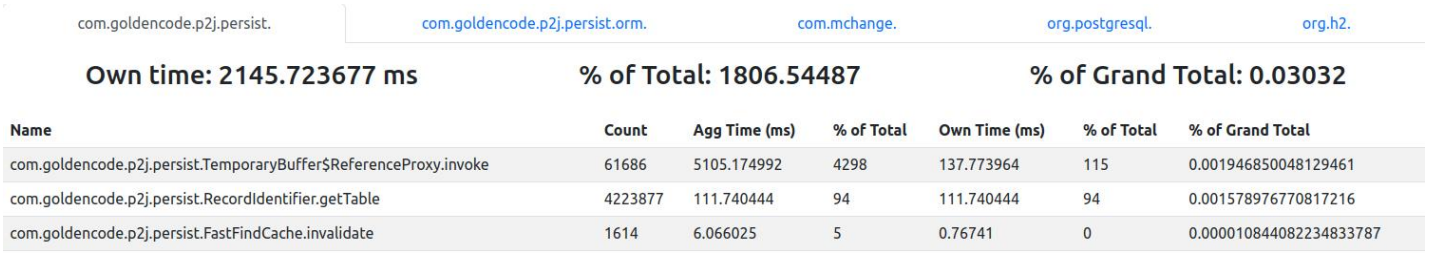
Bucket	Agg Own Time (nano)	% of Grand Total
com.goldencode.p2j.persist.	70444817414	0.683147461116066
com.goldencode.p2j.persist.orm.	9293302474	0.090122967502149
com.mchange.	1309370484	0.012697784658139
org.postgresql.	5214145919	0.050564834677897
org.h2.	36052905198	0.349627574547125

Note that the percentages add up to >100%. This defect probably dates back to my original implementation. Can you please find and fix the problem?

#31 - 10/27/2022 02:55 AM - Eric Faulhaber

- File 4056_bucket.png added

There are also some cases where a method's aggregate and own times are reported as >100% of the bucket total:



#32 - 10/27/2022 04:06 AM - Alexandru Lungu

The aggregate times per bucket are not displayed yet; working on it.
I will check how the sums are computed (and the totals) to ensure correctness.

I have a slight issue right now. I can't test Hotel GUI with 6129b because:

```
com.goldencode.p2j.persist.PersistenceException: Failed to execute [SELECT COUNT(*) FROM meta_user WHERE upper
(rtrim(_userid)) = upper(rtrim(?)) AND _password = ?]
Caused by: org.postgresql.util.PSQLException: ERROR: column "_userid" does not exist
```

```
Hint: Perhaps you meant to reference the column "meta_user.userid".
```

Maybe I should change something in my Hotel GUI project to make it compatible with 6129b? If this is too complex, I can continue developing on another branch and commit my changes to 6129b, as they are not that intrusive to require branch-specific testing.

#33 - 10/27/2022 10:26 AM - Alexandru Lungu

Extended the method tracing with a new file: summary_trace.csv including

- Summary table of scenarios: root, count, own time, % of time. Also attached own time and % of time for each package used from the root.
- A list of all methods called by the tracing with count and own time.
- A list of grand totals (*fixed)
- No negative own times are generated: this was caused by closing method tracing before methods stop. In this case, the sub-calls were sub-tracing time from a not-ended method (which has total time 0). Fixed this by presuming for this scenarios that the own time is 0, so the total time equals the time of the sub-calls.
- Grand total is computed in regard to the profiled packages and not the total elapsed time. There may be a package which is not included by the method tracing. Therefore, some times are not captured by the CSV. Can't detect very easily which packages are used but not captured.
- The total time of traced methods equal the total time spent in the buckets. The sum of the aggregate time however is a bit lower - investigating.

#34 - 10/27/2022 11:49 AM - Constantin Asofiei

I've fixed the issue with the wrong times - the thread-local must be static, for it to work as 'thread-local'. See 6129b/14312.

#35 - 11/02/2022 02:23 PM - Constantin Asofiei

I've committed to 6129b/14318 the changes in DatasetWrapper (serialization ID) and SortIndex (made the class public), which were needed for some cases of the AOP method tracing to work.

Files

trace.csv	107 KB	06/10/2019	Eric Faulhaber
trace.csv	246 KB	10/19/2022	Eric Faulhaber
4056_bucket.png	89.3 KB	10/27/2022	Eric Faulhaber