Database - Bug #6767

Possible name clashes in DMO classes

09/19/2022 08:12 AM - Vladimir Tsichevski

Status:	Review	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Vladimir Tsichevski	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD		

Description

History

#2 - 09/19/2022 09:03 AM - Vladimir Tsichevski

In DMO classes, getter and setter names are created automatically by conversion, based on the field names which are application-specific, so they can be anything matching getter or setter creation rules.

There is possibility that these automatically created names may shadow method names in FWD runtime library, which will lead to runtime errors, which is hard to identify. The example is #6694 issue.

In this task, all such possible name conflicts needs to be identified, and methods, defined in FWD runtime, renamed to names, which are do not match the getter/setter pattern, making the name conflict impossible for any converted customer code.

#3 - 09/19/2022 06:43 PM - Eric Faulhaber

To clarify the task, we need to work through all superinterfaces of the persist.Buffer interface, along with all the ancestors of those superinterfaces, to make sure none of the method names defined in that hierarchy of interfaces start with get, is, or set.

The typical convention we use to prevent this conflict is to use just the base name of the property represented by the method as the method name. For example, consider an interface named Foo, which allows getting and setting properties named bar and baz. Assume bar is a boolean property and baz is a String property. Today, the interface Foo likely defines these methods:

```
public boolean isBar();
public void setBar(boolean bar);
public String getBaz();
public void setBaz(String baz);
```

These would be changed to:

```
public boolean bar();
public void bar(boolean bar);
public String baz();
public void baz(String baz);
```

Any such changes need to be worked back through 4GL method conversion rules, to make sure the changes are reflected there as well.

#4 - 09/22/2022 10:16 AM - Vladimir Tsichevski

Eric Faulhaber wrote:

> To clarify the task, we need to work through all superinterfaces of the persist.Buffer interface, along with all the ancestors of those superinterfaces, to make sure none of the method names defined in that hierarchy of interfaces start with get, is, or set.

Are we considering getters/setters only, or **any** methods with matching names? For example, need we fix methods like this: character getCallbackProcName(String)?

#5 - 09/22/2022 11:05 AM - Vladimir Tsichevski

Here is the list of matching by name methods:

```
character getADMData()
character getCallbackProcName(String)
character getCallbackProcName(Text)
character getDbName()
character getSerializeName()
character getTable()
handle getCallbackProcContext(String)
handle getCallbackProcContext(Text)
handle getQueryAsHandle()
integer getUniqueID()
logical getChanges(handle)
logical getChanges(handle,boolean)
logical getChanges(handle,logical)
logical isDataSourceModified()
logical isMultiTenant()
logical setCallback(String,String)
logical setCallback(String,String,handle)
logical setCallback(String,String,object)
logical setCallback(String,Text,handle)
logical setCallbackProcedure(character,character)
logical setCallbackProcedure(character,character,handle)
logical setCallbackProcedure(character,String)
logical setCallbackProcedure(character,String,handle)
logical setCallbackProcedure(String,character)
logical setCallbackProcedure(String,character,handle)
logical setCallbackProcedure(String,String)
logical setCallbackProcedure(String,String,handle)
logical setCallback(Text,Text)
logical setCallback(Text,Text,handle)
logical setCallback(Text,Text,object)
static logical isDataSourceModified(Buffer)
void setADMData(character)
void setADMData(String)
void setCurrentIteration(handle)
void setDataSourceModified(boolean)
void setDataSourceModified(logical)
void setMultiTenant(boolean)
void setMultiTenant(logical)
void setQueryAsHandle(handle)
void setQueryAsHandle(QueryWrapper)
void setSerializeName(String)
void setSerializeName(Text)
```

the true getters/setters are:

```
character getADMData()
character getDbName()
character getSerializeName()
character getTable()
handle getQueryAsHandle()
integer getUniqueID()
logical isDataSourceModified()
logical isMultiTenant()
static logical isDataSourceModified(Buffer)
void setADMData(character)
void setADMData(String)
void setCurrentIteration(handle)
void setDataSourceModified(boolean)
void setDataSourceModified(logical)
void setMultiTenant(boolean)
void setMultiTenant(logical)
void setQueryAsHandle(handle)
void setQueryAsHandle(QueryWrapper)
void setSerializeName(String)
void setSerializeName(Text)
```

#6 - 10/03/2022 09:04 AM - Roger Borrello

- Related to Bug #6509: Update to Keikai 5.10.0 added

#7 - 10/03/2022 09:05 AM - Roger Borrello

- Related to deleted (Bug #6509: Update to Keikai 5.10.0)

#8 - 10/04/2022 02:38 PM - Eric Faulhaber

Vladimir Tsichevski wrote:

> Eric Faulhaber wrote:
>
> > To clarify the task, we need to work through all superinterfaces of the persist.Buffer interface, along with all the ancestors of those superinterfaces, to make sure none of the method names defined in that hierarchy of interfaces start with get, is, or set.
>
> Are we considering getters/setters only, or **any** methods with matching names? For example, need we fix methods like this: character getCallbackProcName(String)?

Sorry, I missed this question earlier.

DMO getters/setters can be of the form:

```
// scalar field methods
<BDT>   get<Name>()
logical is<Name>()
void    set<Name>(<BDT>)

// extent field methods
<BDT>   get<Name>(int)
<BDT>   get<Name>(NumberType)
logical is<Name>(int)
logical is<Name>(NumberType)
void    set<Name>(int, <BDT>)
void    set<Name>(NumberType, <BDT>)

// special bulk access extent field methods
<BDT>[] get<Name>()
void    set<Name>(<BDT>)
void    set<Name>(<BDT>[])
```

Where <BDT> represents a BaseDataType subclass.

Any signatures which can conflict with these forms should be replaced.

#9 - 10/10/2022 09:15 AM - Vladimir Tsichevski

Eric Faulhaber wrote:

> DMO getters/setters can be of the form:
>
> [...]

So, the following methods from the list in #6767-5 do **match** the criteria, and must be renamed:

```
character getADMData()
character getDbName()
character getSerializeName()
character getTable()
handle getQueryAsHandle()
integer getUniqueID()
logical isDataSourceModified()
logical isMultiTenant()
void setADMData(character)
void setCurrentIteration(handle)
void setDataSourceModified(logical)
void setMultiTenant(logical)
void setQueryAsHandle(handle)
void setSerializeName(Text)
```

and these do **not** match and should be left intact:

```
character getCallbackProcName(String)
character getCallbackProcName(Text)
handle getCallbackProcContext(String)
handle getCallbackProcContext(Text)
logical getChanges(handle)
logical getChanges(handle,boolean)
logical getChanges(handle,logical)
static logical isDataSourceModified(Buffer)
logical setCallback(String,String)
logical setCallback(String,String,handle)
logical setCallback(String,String,object)
logical setCallback(String,Text,handle)
logical setCallbackProcedure(character,character)
logical setCallbackProcedure(character,character,handle)
logical setCallbackProcedure(character,String)
logical setCallbackProcedure(character,String,handle)
logical setCallbackProcedure(String,character)
logical setCallbackProcedure(String,character,handle)
logical setCallbackProcedure(String,String)
logical setCallbackProcedure(String,String,handle)
logical setCallback(Text,Text)
logical setCallback(Text,Text,handle)
logical setCallback(Text,Text,object)
void setADMData(String)
void setDataSourceModified(boolean)
void setMultiTenant(boolean)
void setQueryAsHandle(QueryWrapper)
void setSerializeName(String)
```

#10 - 10/10/2022 06:23 PM - Vladimir Tsichevski

- Status changed from New to WIP

- File 6767.diff added

- % Done changed from 0 to 100

Attached is the patch for this issue (6767.diff).

After this patch is applied, you will need to re-convert all customer application, or to do a series of global replacements in converted Java code, and recompile:

```
replace-all.sh 'isDataSourceModified' 'dataSourceModified'
replace-all.sh 'isMultiTenant' 'multiTenant'
replace-all.sh 'setCurrentIteration' 'currentIteration'
replace-all.sh 'setDataSourceModified' 'dataSourceModified'
replace-all.sh 'setMultiTenant' 'multiTenant'

replace-all.sh 'setSerializeName' 'serializeName'
replace-all.sh 'unwrapQueryAssociable().getQueryAsHandle' 'unwrapQueryAssociable().queryAsHandle'
replace-all.sh 'unwrapUniqueID().getUniqueID' 'unwrapUniqueID().uniqueID'
replace-all.sh 'unwrapADMData().setADMData' 'unwrapADMData().admData'
replace-all.sh 'unwrapADMData().getADMData' 'unwrapADMData().admData'
replace-all.sh 'unwrapDatabaseInfo().getTable' 'unwrapDatabaseInfo().table'
replace-all.sh 'setQueryAsHandle' 'queryAsHandle'
replace-all.sh 'getQueryAsHandle' 'queryAsHandle'
replace-all.sh 'unwrapDatabaseInfo().getDbName' 'unwrapDatabaseInfo().dbName'
```

Here replace-all.sh is a script I use to do global replacements.

#11 - 10/10/2022 06:23 PM - Vladimir Tsichevski

- Status changed from WIP to Review

- Assignee set to Vladimir Tsichevski

#12 - 10/10/2022 06:32 PM - Vladimir Tsichevski

- % Done changed from 100 to 80

- Status changed from Review to WIP

UPD: this patch is probably causes some regressions like org.postgresql.util.PSQLException: ERROR: column wfnregio__0_.bgcnr does not exist. Investigating...

#13 - 10/11/2022 08:05 AM - Vladimir Tsichevski

Some more changes needed, since some TRPL properties do not exist anymore, and the method call must be used now instead of property name:

```
=== modified file 'rules/annotations/record_scoping.rules'
--- rules/annotations/record_scoping.rules    2022-06-27 00:16:13 +0000
+++ rules/annotations/record_scoping.rules    2022-10-11 11:40:27 +0000
@@ -455,7 +455,7 @@
                <action>scope.putAnnotation("is_dynamic_table", true)</action>
             </rule>
             <action>scope.putAnnotation("schemaname", master.schemaName)</action>
-            <action>scope.putAnnotation("dbname", master.dbName)</action>
+            <action>scope.putAnnotation("dbname", master.dbName())</action>
             <action>scope.putAnnotation("scopetype", #(long) bufscope.type)</action>
             <action>scope.putAnnotation("implicit", master.implicit and !master.inSuper)</action>
             <action>scope.putAnnotation("readOnly", master.readOnly)</action>

=== modified file 'rules/convert/builtin_functions.rules'
--- rules/convert/builtin_functions.rules    2022-10-05 08:32:33 +0000
+++ rules/convert/builtin_functions.rules    2022-10-11 11:44:45 +0000
@@ -578,7 +579,7 @@

          <!-- DATA-SOURCE-MODIFIED function -->
          <rule>ftype == prog.kw_data_sm
-            <action>methodText = "DataSourceModifiable.isDataSourceModified"</action>
+            <action>methodText = "DataSourceModifiable.dataSourceModified()"</action>
             <action>dbimport = true</action>
          </rule>
```

#14 - 10/11/2022 08:07 AM - Vladimir Tsichevski

- Status changed from WIP to Review

- % Done changed from 80 to 100

Vladimir Tsichevski wrote:

> UPD: this patch is probably causes some regressions like org.postgresql.util.PSQLException: ERROR: column wfnregio__0_.bgcnr does not exist.
> Investigating...

False alarm: the problem was due to incompatible customer binary release was used.
Now the issue seems to be resolved, please review.

#15 - 10/11/2022 08:16 AM - Greg Shah

Please post the final proposed patch.

#16 - 10/11/2022 08:28 AM - Vladimir Tsichevski

- File 6767-full.diff added

Greg Shah wrote:

> Please post the final proposed patch.

Done 6767-full.diff.

#17 - 10/14/2022 05:04 PM - Greg Shah

Eric: Please review.

#18 - 12/09/2022 10:30 AM - Greg Shah

Eric: This is still waiting for review.

#19 - 03/29/2023 12:58 PM - Eric Faulhaber

Code review 6767-full.diff:

The changes look good.

Just some minor issues with the file headers:

- The dates are set to "2022yyyy". Regardless of the date and order of when these changes get committed, the date should reflect when the work was done.
- Older entries can be changed for format or to correct typos, but the content should not be altered to reflect later changes to the code. For example, in BufferImpl.java, the 20131128 entry should not be updated to reflect the new name of the uniqueID method. It was named getUniqueID at that time and should be left that way, since the entries are meant to be historical.

#20 - 03/29/2023 05:10 PM - Vladimir Tsichevski

Eric Faulhaber wrote:

> Code review 6767-full.diff:
>
> The changes look good.
>
> Just some minor issues with the file headers:
>
> - The dates are set to "2022yyyy". Regardless of the date and order of when these changes get committed, the date should reflect when the work was done.

Got it. I will put today's date in this very case. From now on will be follow your rules.

- Older entries can be changed for format or to correct typos, but the content should not be altered to reflect later changes to the code. For example, in BufferImpl.java, the 20131128 entry should not be updated to reflect the new name of the uniqueID method. It was named getUniqueID at that time and should be left that way, since the entries are meant to be historical.

That is understandable. History can only be rewritten in Orwell's novels or here in Russia (where I fill myself living in the "1984" :-( )

What am I expected to do to make this patch merged to the trunk?

#21 - 03/30/2023 03:05 AM - Eric Faulhaber

Vladimir Tsichevski wrote:

> What am I expected to do to make this patch merged to the trunk?

Create a task branch 6767a, apply the diffs, and we'll follow the usual procedure. This has lower priority than #3827.

#22 - 03/31/2023 07:28 PM - Vladimir Tsichevski

Eric Faulhaber wrote:

> Vladimir Tsichevski wrote:
>
> > What am I expected to do to make this patch merged to the trunk?
>
> Create a task branch 6767a, apply the diffs, and we'll follow the usual procedure. This has lower priority than #3827.

Done. Revno is 14522. Also some minor history issues cleaned up. Please, review.

#23 - 09/15/2023 10:30 AM - Greg Shah

Eric: Please review.

#24 - 12/20/2023 04:39 PM - Eric Faulhaber

Vladimir, sorry for the long delay in reviewing this again. Please rebase this branch when you can. I started reviewing 6767a/14522, but I suspect some of the changes may already be in trunk in some form, due to bugs having been found/fixed in the time since this branch was created.

#25 - 12/21/2023 08:42 AM - Vladimir Tsichevski

Eric Faulhaber wrote:

> Vladimir, sorry for the long delay in reviewing this again. Please rebase this branch when you can. I started reviewing 6767a/14522, but I suspect some of the changes may already be in trunk in some form, due to bugs having been found/fixed in the time since this branch was created.

The branch is ~370 releases old, rebasing it causes many conflicts. I think, it is simpler to redo the task on the latest FWD release now than resolving all the conflicts manually :-(

#26 - 12/22/2023 03:01 PM - Vladimir Tsichevski

The following methods should be also renamed in addition to the methods listed in #6767-9:

```
handle getIteration(int)
handle getIteration(NumberType)
character getXmlDataType()
setXmlDataType(Text)
character getXmlNodeType()
setXmlNodeType(Text)
character getXmlNodeName()
setXmlNodeName(Text)
```

The complete method list is:

```
public abstract com.goldencode.p2j.util.logical com.goldencode.p2j.persist.Buffer.isMultiTenant()
public abstract void com.goldencode.p2j.persist.Buffer.setMultiTenant(com.goldencode.p2j.util.logical)
public abstract com.goldencode.p2j.util.character com.goldencode.p2j.util.ADMData.getADMData()
public abstract void com.goldencode.p2j.util.ADMData.setADMData(com.goldencode.p2j.util.character)
public abstract com.goldencode.p2j.util.character com.goldencode.p2j.util.DatabaseInfo.getTable()
public abstract com.goldencode.p2j.util.character com.goldencode.p2j.util.DatabaseInfo.getDbName()
public abstract com.goldencode.p2j.util.logical com.goldencode.p2j.persist.DataSourceModifiable.isDataSourceMo
dified()
public abstract void com.goldencode.p2j.persist.DataSourceModifiable.setDataSourceModified(com.goldencode.p2j.
util.logical)
public abstract void com.goldencode.p2j.util.IterableResource.setCurrentIteration(com.goldencode.p2j.util.hand
le)
public abstract com.goldencode.p2j.util.handle com.goldencode.p2j.util.IterableResource.getIteration(int)
public default com.goldencode.p2j.util.handle com.goldencode.p2j.util.IterableResource.getIteration(com.golden
code.p2j.util.NumberType)
public abstract com.goldencode.p2j.util.character com.goldencode.p2j.persist.NamedSerializable.getSerializeNam
e()
public abstract void com.goldencode.p2j.persist.NamedSerializable.setSerializeName(com.goldencode.p2j.util.Tex
t)
public abstract void com.goldencode.p2j.persist.QueryAssociable.setQueryAsHandle(com.goldencode.p2j.util.handl
e)
public abstract com.goldencode.p2j.util.handle com.goldencode.p2j.persist.QueryAssociable.getQueryAsHandle()
public abstract com.goldencode.p2j.util.integer com.goldencode.p2j.util.UniqueID.getUniqueID()
public abstract com.goldencode.p2j.util.character com.goldencode.p2j.persist.XmlNode.getXmlDataType()
```

```
public abstract void com.goldencode.p2j.persist.XmlNode.setXmlDataType(com.goldencode.p2j.util.Text)
public abstract com.goldencode.p2j.util.character com.goldencode.p2j.persist.XmlNode.getXmlNodeType()
public abstract void com.goldencode.p2j.persist.XmlNode.setXmlNodeType(com.goldencode.p2j.util.Text)
public abstract com.goldencode.p2j.util.character com.goldencode.p2j.persist.XmlNode.getXmlNodeName()
public abstract void com.goldencode.p2j.persist.XmlNode.setXmlNodeName(com.goldencode.p2j.util.Text)
```

#27 - 12/25/2023 07:04 AM - Vladimir Tsichevski

The work redone in the branch 6767b, The branch is rebased to trunk rev. 14906 the branch revision is 14917.
Please, review the cumulative change 14906-14917.

Files

6767.diff	99.2 KB	10/10/2022	Vladimir Tsichevski
6767-full.diff	100 KB	10/11/2022	Vladimir Tsichevski