# Base Language - Feature #2041

## implement runtime support for "shared" attributes that are just stubbed

02/23/2013 11:51 AM - Greg Shah

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 10/23/2013 |
| **Priority:** | Normal | **Due date:** | 11/01/2013 |
| **Assignee:** | Hynek Cihlar | **% Done:** | 100% |
| **Category:** | | **Estimated time:** | 40.00 hours |
| **Target version:** | Runtime Support for Server Features | | |
| **billable:** | No | **version:** | |
| **vendor_id:** | GCD | | |

**Description**

**Related issues:**

| | | |
|---|---|---|
| Related to Base Language - Feature #2002: refactor some "shared" attributes | **Closed** | **02/18/2013** |

## History

**#1 - 04/26/2013 07:42 AM - Greg Shah**

*- Estimated time changed from 24.00 to 40.00*

**#2 - 04/26/2013 09:38 AM - Greg Shah**

*- Due date set to 07/17/2013*

*- Assignee set to Greg Shah*

*- Start date set to 07/06/2013*

**#3 - 04/26/2013 09:45 AM - Greg Shah**

*- Due date changed from 07/17/2013 to 07/19/2013*

**#4 - 09/24/2013 11:44 AM - Greg Shah**

Constantin: I have looked at #2002 to see the runtime implementation status of these attributes. It is not clear from the history there.

I then started looking at the runtime code and there is definitely some implementation of most of these attributes. For example, it seems like NAME, TYPE and PRIVATE-DATA have default implementations for most resources (in HandleResource and HandleChain). But it is not clear if there are any missing resources or remaining work on these attributes.

I need to know the state of all of the shared attributes listed in #2002. Can you summarize the remaining work/issues?

**#5 - 09/24/2013 01:22 PM - Constantin Asofiei**

Implemented by some resources:
- local-name: x-noderef (pending to be released), soap-header-entryref
- namespace-uri: SOAP-header-entryref, Temp-table, X-document (pending), X-noderef (pending).
For these two, each resource should have its own management.

Not implemented and need generic implementation:
- adm-data, unique-id, instantiating-procedure: there should be a common management for these kind of attributes, as their behavior is the same. If you look in HandleChain.WorkArea, you will see that I use a map to hold the last/first resource in the chain, per resource type. Something similar should be done for these too, and the implementation at the resource class just call the same worker.

COLUMN-LABEL (BUFFER-FIELD), DATA-TYPE, FORMAT, LABEL:
- COLUMN-LABEL is used only by Buffer Field resource
- DATA-TYPE, FORMAT is used by both widgets and Buffer Field

- LABEL is used by widgets, Buffer Field and LAST-EVENT
This is what needs to be done.
- GenericWidget.get/setColumnLabel should call handle.invalidAttribute, and not just throw exception
- DATA-TYPE, FORMAT, LABEL - GenericWidget implements them fully
- Buffer Field resource needs to be implemented AFAIK
- LAST-EVENT:LABEL needs to be implemented

Already implemented:
sensitive: widgets, socket, server-socket all should implement it properly
NAME, PRIVATE-DATA, NEXT-SIBLING, PREV-SIBLING: implemented in HandleChain
type: implemented in HandleResource (and HandleChain extends HandleResource)

As long as the resource is extending the HandleChain class, all is good. And I think all current resources do implement it (Except the ExternalProgramWrapper case, which has its own management, as this is a special case).

**#6 - 09/24/2013 02:06 PM - Greg Shah**

It looks like we have a PRIVATE-DATA implementation in both BaseEntity and QueryWrapper that needs to be deleted.  Both of these derive from HandleChain which already implements the proper methods.

**#7 - 09/24/2013 02:36 PM - Constantin Asofiei**

Greg Shah wrote:

> It looks like we have a PRIVATE-DATA implementation in both BaseEntity and QueryWrapper that needs to be deleted.  Both of these derive from HandleChain which already implements the proper methods.

Yes, they can be removed.

**#8 - 10/20/2013 01:40 PM - Greg Shah**

*- Assignee changed from Greg Shah to Hynek Cihlar*

*- Start date changed from 07/06/2013 to 10/23/2013*

*- Due date changed from 07/19/2013 to 11/01/2013*

The BUFFER-FIELD implementation of COLUMN-LABEL, DATA-TYPE, FORMAT and LABEL will be handled in other tasks that are currently being worked.

LAST-EVENT:LABEL is already being implemented by #2020.

The other changes listed above should be implemented.

**#9 - 11/30/2013 04:45 PM - Hynek Cihlar**

*- Status changed from New to WIP*

**#10 - 11/30/2013 04:50 PM - Hynek Cihlar**

*- File Resource attributes.ods added*

I have summarized the state of all attributes and what needs to be done. Please review the attached Calc sheet.

Also

- XEntityImpl.getName and XEntityImpl.setName are not consistent. The getter reads the name from the related node, however the setter sets the name into the HandleChain.name. Similarly for XDocumentImpl.
- NullWidget throws an exception if any of the attribute getters are called. Why not the setters?

**#11 - 11/30/2013 05:51 PM - Constantin Asofiei**

Hynek, some notes:

1. SAX resource is WIP by Evgeny (and I expect all attrs/methods to be implemented, including ADM-DATA and UNIQUE-ID).
2. in the end, each resource contains its own support for ADM-DATA and UNIQUE-ID (I mean, these are saved in instance fields local for each resource).
3. INSTANTIATING-PROCEDURE is being added by #2196 (is still WIP, hope to have an update soon)
4. XEntityImpl.getName and XEntityImpl.setName - this is a question for Evgeny: is the NAME attribute read-only for X-DOCUMENT/X-NODEREF/etc? If yes, then XEntityImpl should override hasNameReadOnly and let it return false; also, override setName and annotate it accordingly (so that the setter will be ignored).
5. NullWidget is something used internally by P2J to reference "unknown widgets"; at some point we will get rid of it, please ignore it for now.
6. expect that TempTableBuilder (and others) to be refactored/restructured with the #1654 task

**#12 - 12/01/2013 11:27 AM - Hynek Cihlar**

Constantin Asofiei wrote:

> Hynek, some notes:
>
> 1. SAX resource is WIP by Evgeny (and I expect all attrs/methods to be implemented, including ADM-DATA and UNIQUE-ID).

Ok.

> 1. in the end, each resource contains its own support for ADM-DATA and UNIQUE-ID (I mean, these are saved in instance fields local for each resource).

Could you please explain what was meant by "common management" from note-5 above? Progress documentation doesn't indicate these attributes should need any special kind of behavior.

Yes, there's no suitable common parent class for adm-data and unique-id. But the attribute implementations would be trivial so I would not try to create any "smart" common logic. Unless I am missing something here.

> 1. INSTANTIATING-PROCEDURE is being added by #2196 (is still WIP, hope to have an update soon)

Ok, I am excluding this from my list.

1. NullWidget is something used internally by P2J to reference "unknown widgets"; at some point we will get rid of it, please ignore it for now.

Ok.

1. expect that TempTableBuilder (and others) to be refactored/restructured with the #1654 task

Thanks for pointing this out, will watch out.

#### #13 - 12/01/2013 12:08 PM - Hynek Cihlar

SOAPHeaderEntryImpl is heavily unimplemented. Still implement those local-name and namespace-uri attributes mentioned above in isolation? Maybe it would make more sense to implement them together with the rest.

#### #14 - 12/01/2013 01:44 PM - Hynek Cihlar

Buffer resource is not mentioned but is also missing implementations for adm-data, unique-id, namespace-uri and buffer-field. Should I ignore or implement?

#### #15 - 12/01/2013 02:00 PM - Hynek Cihlar

GenericWidget columnLabel setter/getter should call handle.invalidAttribute not just throw an exception (see note-5). There are more attributes handled by only throwing an exception. Should those be also changed in the same way?

#### #16 - 12/01/2013 06:41 PM - Evgeny Kiselev

Constantin Asofiei wrote:

1. XEntityImpl.getName and XEntityImpl.setName - this is a question for Evgeny: is the NAME attribute read-only for X-DOCUMENT/X-NODEREF/etc? If yes, then XEntityImpl should override hasNameReadOnly and let it return false; also, override setName and annotate it accordingly (so that the setter will be ignored).

For X-DOCUMENT/X-NODEREF name is readOnly. For X-NODEREF name takes from node and always readonly too. for some nodes name could be specify in constructor for other nodes (like text, which doesn't have name) name will be node type(#text for text node).
UNIQUE-ID may be grouped, for example for sax-reader, sax-writer, sax-attributes they share same unique sequence. Maybe same idea in other object types exists.

**#17 - 12/02/2013 03:08 AM - Constantin Asofiei**

Hynek Cihlar wrote:

> 1. in the end, each resource contains its own support for ADM-DATA and UNIQUE-ID (I mean, these are saved in instance fields local for each resource).
>
> Could you please explain what was meant by "common management" from note-5 above? Progress documentation doesn't indicate these attributes should need any special kind of behavior.
>
> Yes, there's no suitable common parent class for adm-data and unique-id. But the attribute implementations would be trivial so I would not try to create any "smart" common logic. Unless I am missing something here.

I was thinking to have some common storage for these attributes, for all resources (either in some superclass fields or in a dedicated class, per resource); but, this would only complicate the design, and, as you say, adding support for them at each resource is trivial.

> SOAPHeaderEntryImpl is heavily unimplemented. Still implement those local-name and namespace-uri attributes mentioned above in isolation? Maybe it would make more sense to implement them together with the rest.

SOAP support is not yet started; you can go ahead and add support for ADM-DATA and UNIQUE-ID (as these are trivial), but LOCAL-NAME and NAMESPACE-URI I think is best to leave them to be implemented when SOAP resources are implemented (as you will not be able to test the behaviour without full support).

> Buffer resource is not mentioned but is also missing implementations for adm-data, unique-id, namespace-uri and buffer-field. Should I ignore or implement?

Check #1654, BUFFER resource is part of that task. Stanislav: did you add support for these?

> GenericWidget columnLabel setter/getter should call handle.invalidAttribute not just throw an exception (see note-5). There are more attributes handled by only throwing an exception. Should those be also changed in the same way?

First, please double check in 4GL that indeed these attributes are not supported.

**#18 - 12/05/2013 03:24 PM - Hynek Cihlar**

Evgeny Kiselev wrote:

> Constantin Asofiei wrote:
>
> > 1. XEntityImpl.getName and XEntityImpl.setName - this is a question for Evgeny: is the NAME attribute read-only for X-DOCUMENT/X-NODEREF/etc? If yes, then XEntityImpl should override hasNameReadOnly and let it return false; also, override setName and annotate it accordingly (so that the setter will be ignored).
>
> For X-DOCUMENT/X-NODEREF name is readOnly. For X-NODEREF name takes from node and always readonly too. for some nodes name could be specify in constructor for other nodes (like text, which doesn't have name) name will be node type(#text for text node).

Evgeny, do you want me to change it? Just let me know, it should be no problem.

> UNIQUE-ID may be grouped, for example for sax-reader, sax-writer, sax-attributes they share same unique sequence. Maybe same idea in other object types exists.

This is probably out of scope of this issue.

**#19 - 12/05/2013 03:33 PM - Hynek Cihlar**

Constantin Asofiei wrote:

> Hynek Cihlar wrote:
>
> > SOAPHeaderEntryImpl is heavily unimplemented. Still implement those local-name and namespace-uri attributes mentioned above in isolation? Maybe it would make more sense to implement them together with the rest.
>
> SOAP support is not yet started; you can go ahead and add support for ADM-DATA and UNIQUE-ID (as these are trivial), but LOCAL-NAME and NAMESPACE-URI I think is best to leave them to be implemented when SOAP resources are implemented (as you will not be able to test the behaviour without full support).

Ok, will add support for ADM-DATA and UNIQUE-ID and will leave the rest out.

> > Buffer resource is not mentioned but is also missing implementations for adm-data, unique-id, namespace-uri and buffer-field. Should I ignore or implement?
>
> Check #1654, BUFFER resource is part of that task.  Stanislav: did you add support for these?

The latest code changes from #1654 are missing the implementations for the above-mentioned attributes. Stanislav, are you planning to implement them?

> > GenericWidget columnLabel setter/getter should call handle.invalidAttribute not just throw an exception (see note-5). There are more attributes handled by only throwing an exception. Should those be also changed in the same way?
>
> First, please double check in 4GL that indeed these attributes are not supported.

Ack, I will post the findings first.

**#20 - 12/05/2013 03:44 PM - Stanislav Lomany**

> Stanislav, are you planning to implement them?

There are ADM-DATA and UNIQUE-ID attributes for tables. I'm not planning to implement them for buffers.

**#21 - 12/08/2013 04:52 PM - Hynek Cihlar**

*- File Widget attributes.ods added*

Hynek Cihlar wrote:

> Constantin Asofiei wrote:
>
> > Hynek Cihlar wrote:
> >
> > > GenericWidget columnLabel setter/getter should call handle.invalidAttribute not just throw an exception (see note-5). There are more attributes handled by only throwing an exception. Should those be also changed in the same way?
> >
> > First, please double check in 4GL that indeed these attributes are not supported.
>
> Ack, I will post the findings first.

I am attaching "Widget attributes.ods" describing the current state of unimplemented (or partially unimplemented) widget attributes. Please review and identify attributes to scope in to this issue.

Some of the attributes in the sheet are clearly out of scope (like the ones for pixel-precision gui), I included them anyway for completeness. There are number of unimplemented methods in the source code, I ignored those.

**#22 - 12/10/2013 02:34 PM - Constantin Asofiei**

From this list, only the DATA-TYPE and TABLE attributes are used in appserver reachable code; and, all of them are used with the BUFFER-field or BUFFER resource.

Greg: for M7, my suggestion is to add only proper error support for attributes NOT supported by certain widgets (instead of throwing java exceptions). All attributes which are supported by a widget but their implementation throw a java exception should be left untouched (this way, if we missed something which needs to be implemented, it will be obvious in the logs).

**#23 - 12/10/2013 03:58 PM - Greg Shah**

I don't quite understand what the spreadsheet is saying. For example, the DCOLOR attribute is already supported across most (if not all) widgets today. Perhaps there is something missing, but it is not completely unimplemented. Yet the spreadsheet lists it as "not implemented". The GenericWidget exception is really only there such that if we are missing the support in some widget, that we would easily find it. But so long as the subclasses implement the support properly, this error can never be thrown. Before we go too far with a discussion, I'd like to understand what is being proposed here.

**#24 - 12/10/2013 04:28 PM - Constantin Asofiei**

Greg: the list is a break-down of widget attributes by their associated widgets, but, as you note, the "P2J support" assumptions are not necessarily correct. And to rephrase my note, what I was saying was that the default implementation in GenericWidget/BaseEntity should not be "throw exception"; it should be a 4GL-style error, as i.e. QUERY is supported only by BROWSE widget; and, as in 4GL GenericWidget is the root class for all widgets, this will allow access to the QUERY attribute for non-BROWSE widgets (this should not be possible).

Some additional notes: from the list at note 3, work should be done at this task only for UNIQUE-ID, ADM-DATA, NAMESPACE-URI and LOCAL-NAME attributes - make sure all resources which are already implemented in P2J implement them properly. They should be implemented now only if the attribute implementation is not resource specific.

The implementation for the remainder of those mentioned at note 3 is very resource specific and there is no reason to implement them here.

**#25 - 12/10/2013 04:34 PM - Greg Shah**

Agreed.

**#26 - 12/11/2013 10:46 AM - Hynek Cihlar**

Greg Shah wrote:

> I don't quite understand what the spreadsheet is saying. For example, the DCOLOR attribute is already supported across most (if not all) widgets today. Perhaps there is something missing, but it is not completely unimplemented. Yet the spreadsheet lists it as "not implemented".

In this particular case DCOLOR getter is not implemented.

> The GenericWidget exception is really only there such that if we are missing the support in some widget, that we would easily find it. But so long as the subclasses implement the support properly, this error can never be thrown. Before we go too far with a discussion, I'd like to understand what is being proposed here.

Yes, I was accounting for this. In case the subclasses (for ALL needed widgets) properly implement the attributes I don't mention the attribute in the table.

**#27 - 12/11/2013 11:18 AM - Hynek Cihlar**

Constantin Asofiei wrote:

> Greg: the list is a break-down of widget attributes by their associated widgets, but, as you note, the "P2J support" assumptions are not

> necessarily correct.

I created the table based on the answer from Constantin at the note 17. I noticed that there are more unimplemented (or wrongly implemented) attributes in the GenericWidget class hierarchy. I raised the concern and interpreted the answer such that I should do more investigation on these suspicious attributes.

The table shows the expected 4GL state (column "4GL support) and the actual P2J state (column "P2J support") and the work delta to bring the expected state into the actual state (column P2J TBD). I quickly went through the data again and don't think there are many wrong assumptions. The only one I see is that the DCOLOR mentions the setter as not implemented while only the getter is not.

**#28 - 12/11/2013 11:28 AM - Greg Shah**

> Some additional notes: from the list at note 3, work should be done at this task only for UNIQUE-ID, ADM-DATA, NAMESPACE-URI and LOCAL-NAME attributes - make sure all resources which are already implemented in P2J implement them properly. They should be implemented now only if the attribute implementation is not resource specific.

Please focus on this part.

**#29 - 12/15/2013 07:54 PM - Hynek Cihlar**

*- File hc_upd20131215a.zip added*

I am attaching the implementation involving ADM-DATA, UNIQUE-ID, PRIVATE-DATA and COLUMN-LABEL attribute. Regression test is in progress. Please review.

Any other attributes to be implemented?

**#30 - 12/16/2013 04:20 AM - Constantin Asofiei**

Hynek, changes look good, nice catch with the UNIQUE-ID/ADM-DATA for a procedure resource (that it should not be saved at the ExternalProgramWrapper instance). Please also move to the ProcedureManager$ProcedureData the ExternalProgramWrapper.instProc field (I should have remembered not to add persistent data to the ExternalProgramWrapper class); the logic in the ExternalProgramWrapper c'tor remains the same, just save it at the associated ProcedureManager$ProcedureData instance.
Also, some notes about the update:

- AbstractTempTable.getADMData - as BDT types are mutable, never expose the private reference to the outside world. always return a new instance.
- add Override annotations to the AbstractTempTable.getADMData and AbstractTempTable.setADMData (make sure all the APIs you've changed/added have this annotation, when appropriate).
- some files are missing history entries
- remove the SAX_ATTRIBUTES/SAX_READER changes - this will be added by #1641 (more, the UNIQUE-ID "seed" is the same for all SAX resources).

**#31 - 12/16/2013 01:38 PM - Hynek Cihlar**

Constantin Asofiei wrote:

> Hynek, changes look good, nice catch with the UNIQUE-ID/ADM-DATA for a procedure resource (that it should not be saved at the ExternalProgramWrapper instance). Please also move to the ProcedureManager$ProcedureData the ExternalProgramWrapper.instProc field (I should have remembered not to add persistent data to the ExternalProgramWrapper class); the logic in the ExternalProgramWrapper c'tor remains the same, just save it at the associated ProcedureManager$ProcedureData instance.

Will do.

> Also, some notes about the update:
>
> - AbstractTempTable.getADMData - as BDT types are mutable, never expose the private reference to the outside world. always return a new instance.

Yes, very true. Actually I moved the getter/setter implementations from TempTableBuilder and missed the simple return.

Will fix.

> - add Override annotations to the AbstractTempTable.getADMData and AbstractTempTable.setADMData (make sure all the APIs you've changed/added have this annotation, when appropriate).

Same as above. Will fix.

> - some files are missing history entries
> - remove the SAX_ATTRIBUTES/SAX_READER changes - this will be added by #1641 (more, the UNIQUE-ID "seed" is the same for all SAX resources).

Ok, will fix.

**#32 - 12/19/2013 05:41 PM - Hynek Cihlar**

*- File hc_upd20131219a.zip added*

Changes attached, please review. Regression test is pending.

**#33 - 12/20/2013 06:37 AM - Constantin Asofiei**

Review of 1219a.zip:

1. I would prefer to have a ProcedureManager.setInstantiatingProcedure(Object referent, handle instProc) instead of ProcedureManager.assignThisProcedureToInstantiatingProcedure. The code in ExternalProgramWrapper c'tor will look like:

   ```
   ProcedureManager.setInstantiatingProcedure(referent, ProcedureMangager.thisProcedure);
   ```

2. leave ProcedureManager.resolveInstantiatingProcedure as is and add a ProcedureManager.getInstantiatingProcedure(Object referent). The implementation of ProcedureManager.resolveInstantiatingProcedure is needed as is. The code in ExternalProgramWrapper.instantiatingProcedure will look like:

   ```
   handle instProc = ProcedureManager.getInstantiatingProcedure(referent);
   return ProcedureManager.resolveInstantiatingProcedure(instProc);
   ```

**#34 - 12/20/2013 06:38 AM - Constantin Asofiei**

PS: the ProcedureManager.getInstantiatingProcedure(Object referent); will perform no validation, will return the handle as it was saved.

**#35 - 12/22/2013 07:08 PM - Hynek Cihlar**

*- File hc_upd20131222a.zip added*

Fixed the issues with the INSTANTIATING-PROCEDURE attribute. Also resolved an incoming conflict on the BufferImpl.getUniqueID. Changes attached, please review.

**#36 - 12/23/2013 02:32 AM - Constantin Asofiei**

1222a.zip looks good, go ahead and get it regression tested.

**#37 - 01/01/2014 05:17 PM - Hynek Cihlar**

*- File hc_upd20140101a.zip added*

Calling ProcedureManager.setInstantiatingProcedure from the constructor of ExternalProgramWrapper is too early, at this time the ProcedureData instance for referent is not created yet. So I moved setting the INSTANTIATING-PROCEDURE to ProcedureManager.WorkArea.scopeStart. I am regression testing the change and for now it seems to work ok. Please review the attached hc_upd20140101a.zip.

**#38 - 01/03/2014 04:34 AM - Constantin Asofiei**

OK, the location is the right one, but you need to do it only if we are starting an external proc; also, remove the ProcedureManager.setInstantiatingProcedure and set the field directly:

```
            // setting INSTANTIATING-PROCEDURE can be done only after ProcedureData is created
            if (bt.equals(BlockType.EXTERNAL_PROC) && TransactionManager.getNestingLevel() > 2)
            {
               // if this is not the entry point, set the INSTANTIATING-PROCEDURE as the current 
               // THIS-PROCEDURE
               pdata.instProc.assign(ProcedureManager.thisProcedure());
            }
```

**#39 - 01/06/2014 07:40 AM - Hynek Cihlar**

*- File hc_upd20140105a.zip added*

Fixed and changes attached. Regression tests passed.

**#40 - 01/06/2014 07:44 AM - Hynek Cihlar**

*- Status changed from WIP to Review*

**#41 - 01/06/2014 07:49 AM - Constantin Asofiei**

OK, remove the ProcedureManager.setInstantiatingProcedure (because is no longer used and it will just confuse readers) and you can release it.

**#42 - 01/06/2014 05:09 PM - Hynek Cihlar**

*- % Done changed from 0 to 100*

*- File hc_upd20140106a.zip added*

The final change set attached. Committed at revision 10427.

**#43 - 01/06/2014 05:46 PM - Greg Shah**

*- Status changed from Review to Closed*

**#44 - 11/16/2016 11:42 AM - Greg Shah**

*- Target version changed from Milestone 7 to Runtime Support for Server Features*

## Files

| | | | |
|---|---|---|---|
| Resource attributes.ods | 14.8 KB | 11/30/2013 | Hynek Cihlar |
| Widget attributes.ods | 17.2 KB | 12/08/2013 | Hynek Cihlar |
| hc_upd20131215a.zip | 97 KB | 12/16/2013 | Hynek Cihlar |
| hc_upd20131219a.zip | 92.6 KB | 12/19/2013 | Hynek Cihlar |
| hc_upd20131222a.zip | 92.7 KB | 12/23/2013 | Hynek Cihlar |
| hc_upd20140101a.zip | 92.4 KB | 01/01/2014 | Hynek Cihlar |
| hc_upd20140105a.zip | 92.4 KB | 01/06/2014 | Hynek Cihlar |
| hc_upd20140106a.zip | 92.4 KB | 01/06/2014 | Hynek Cihlar |