Base Language - Bug #7193

READ-JSON is not supported for the "LONGCHAR" source type

03/14/2023 07:04 AM - Vladimir Tsichevski

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

Description

Related issues:

Related to Database - Feature #6444: dataset improvements	**Closed**	
Related to Conversion Tools - Feature #6237: OEUnit support	**Closed**	
Related to Base Language - Bug #7247: Dataset:READ-XML does nothing	**Closed**	
Related to Database - Bug #4071: FWD server stops allowing new web client log...	**New**	
Related to Base Language - Bug #7345: The memptr deallocation attempt after t...	**New**	**05/11/2023**
Related to Database - Bug #6475: double-check and fix any read/write-xml/json...	**New**	

History

#1 - 03/14/2023 07:23 AM - Igor Skornyakov

Actually READ-JSON is not supported for "file" source type as well.
For TEMP-TABLE it explicitly reports error 19079 file argument for READ-JSON is not valid.
For DATA-SET it silently does nothing,

#2 - 03/14/2023 07:23 AM - Igor Skornyakov

- Related to Feature #6444: dataset improvements added

#3 - 03/14/2023 07:29 AM - Vladimir Tsichevski

- Related to Feature #6237: OEUnit support added

#4 - 03/14/2023 07:31 AM - Vladimir Tsichevski

Igor Skornyakov wrote:

> Actually READ-JSON is not supported for "file" source type as well.
> For TEMP-TABLE it explicitly reports error 19079 file argument for READ-JSON is not valid.
> For DATA-SET it silently does nothing,

Thanks Igor.

I created this task since the DataProvider feature of OEUnit (see #6237) depends on JSON serialization.

#5 - 03/14/2023 09:26 AM - Greg Shah

Are these testcases checked in to the Testcases project? (this is a question for both of you)

#6 - 03/14/2023 09:30 AM - Vladimir Tsichevski

Greg Shah wrote:

> Are these testcases checked in to the Testcases project? (this is a question for both of you)

The DataProvider class uses all kinds of JSON serialization, and the code is going to be part of the FWD project. I can add unit tests for all DataProvider import methods to testcases, but most of them will fail due to this issue.

#7 - 03/14/2023 09:35 AM - Igor Skornyakov

Greg Shah wrote:

> Are these testcases checked in to the Testcases project? (this is a question for both of you)

I do not think that any special testacases are required since READ-JSON doesn't work at all even for very basic tests.
In fact I started to look at this working on #6444 and my test was about support of the DATA-RELATION:FOREIGN-KEY-HIDDEN attribute.

#8 - 03/14/2023 09:35 AM - Greg Shah

That is OK if they fail. We should fix this now.

#9 - 03/14/2023 09:36 AM - Greg Shah

Igor Skornyakov wrote:

> Greg Shah wrote:
>
> > Are these testcases checked in to the Testcases project? (this is a question for both of you)
>
> I do not think that any special testacases are required since READ-JSON doesn't work at all even for very basic tests.
> In fact I started to look at this working on #6444 and my test was about support of the DATA-RELATION:FOREIGN-KEY-HIDDEN attribute.

Then we must already have testcases that fail. Point us to the code that fails.

#10 - 03/14/2023 09:52 AM - Igor Skornyakov

- File json.p added

Greg Shah wrote:

> Then we must already have testcases that fail. Point us to the code that fails.

The attached is the excerpt from my test for #6444 with JSON-related part only.

#11 - 03/16/2023 02:07 AM - Igor Skornyakov

- Assignee set to Igor Skornyakov

- Status changed from New to WIP

Created task branch 7193a from the trunk.

#12 - 03/25/2023 08:53 AM - Igor Skornyakov

Fixed source validation for READ-JSON to work with MEMPTR, LONGCHAR and FILE for TEMP-TABLE and rejects STREAM as a source in correct way. Implemented READ-JSON support for dynamic TEMP-TABLE right after CREATE TEMP-TABLE
Committed to 7193a/14507.

#13 - 03/26/2023 01:15 PM - Igor Skornyakov

- File ds.1.fk.json added

- File json-fk.p added

- File json.p added

- File ds.1.json added

- File ds.json added

I'm working now on READ-JSON for relations with FOREIGN-KEY-HIDDEN.
Please note that the support for this is broken in 4GL
Consider the attached tests. The only difference is that in one the DATA-RELATION are declared with FOREIGN-KEY-HIDDEN while in another one w/o it.
The ds.1.json is the output of WRITE-JSON after DATA-SET ds was exported to JSON and re-imported.
As one can see with FOREIGN-KEY-HIDDEN the result is different of the initial export while w/o it they are identical.
This is a bug since the only reason for FOREIGN-KEY-HIDDEN it to reduce the size of WRITE-XML/WRITE-JSON output and for WRITE-XML this is indeed the fact.

It is unclear however what exactly is broken - READ-JSON or WRITE-JSON.
I've not analized this since it seem to be not very important.

#14 - 03/27/2023 11:59 AM - Igor Skornyakov

Fixed READ-JSON for static DATA-SET with NESTED/FOREIGN-KEY-HIDDEN relations.
Committed to 7193a/14508.

Working in READ-JSON support for just created dynamic data-sets (this is the only remaining thing to be done).

#15 - 04/03/2023 04:42 PM - Vladimir Tsichevski

- Related to Bug #7247: Dataset:READ-XML does nothing added

#16 - 04/03/2023 04:43 PM - Vladimir Tsichevski

Igor, would you take a look at #7247, please. I think, both issues are closely related.

#17 - 04/03/2023 04:53 PM - Igor Skornyakov

Vladimir Tsichevski wrote:

> Igor, would you take a look at #7247, please. I think, both issues are closely related.

Yes. you're rignt (see my comment #7247-4).

#18 - 04/14/2023 01:02 PM - Igor Skornyakov

Inferring empty dynamic DATA-SET from JSON data is mostly finished.
Committed to 7193a/14539.

The tables looks the same as with 4GL. The values of the DATA-SET attributes printed by the test are also identical.
However, the output of the WRITE-JSON and WRITE-XMLSCHEMA for the result of the READ-JSON look different in FWD and 4GL.
This can be the problem with the WRITE-.... methods since I've not tested them with PARENT-ID-RELATION.
Analyzing.

#19 - 04/15/2023 12:19 PM - Igor Skornyakov

I've encountered the following problem.
On inferring DATA-SET from JSON data with READ-JSON the NESTED DATA-RELATION is replaced with PARENT-ID-RELATION. The SERIALIZE-HIDDEN ${parent table name}_id@ field is added to the child table and populated with the RECID of the parent table record.

However in the TempTableSchema c'tor SERIALIZE-HIDDEN fields are ignored (no Column is created) and such fields cannot be populated. This looks strange, but I understand that this was done on purpose.
Any suggestions?
Thank you.

#20 - 04/15/2023 02:31 PM - Igor Skornyakov

- Status changed from WIP to Review

Added support for inferring empty dynamic DATA-SET from JSON data. Fixed support for PARENT-ID-RELATION in WRITE-JSON and WRITE-XMLSCHEMA.
Fixed support for SERIALIZE-HIDDEN in TempTableSchema and WRITE-JSON. (see #7193-19).

Committed to 7193a/14540.
Please review.
Thank you.

#21 - 04/16/2023 02:35 AM - Igor Skornyakov

Igor Skornyakov wrote:

> Added support for inferring empty dynamic DATA-SET from JSON data. Fixed support for PARENT-ID-RELATION in WRITE-JSON and WRITE-XMLSCHEMA.
> Fixed support for SERIALIZE-HIDDEN in TempTableSchema and WRITE-JSON. (see #7193-19).
>
> Committed to 7193a/14540.
> Please review.
> Thank you.

Sorry, I've realized that some minor additional changes are still required. Will be done soon.

#22 - 04/17/2023 03:32 AM - Igor Skornyakov

- File json-fk.p added

Just noticed.
When running the attached test I see multiple message like this

```
Note: 'force_dmo_alias' annotation [tt1] overrides remapping of [tt1Buf2] for buffer [tt-1]
Note: 'force_dmo_alias' annotation [tt2] overrides remapping of [tt2Buf2] for buffer [tt2]
Note: 'force_dmo_alias' annotation [tt3] overrides remapping of [tt3Buf2] for buffer [tt3]
Note: 'force_dmo_alias' annotation [tt4] overrides remapping of [tt4Buf2] for buffer [tt4]
Note: 'force_dmo_alias' annotation [tt5] overrides remapping of [tt5Buf2] for buffer [tt5]
Note: 'force_dmo_alias' annotation [tt5] overrides remapping of [tt5Buf2] for buffer [tt5]
Note: 'force_dmo_alias' annotation [tt1] overrides remapping of [tt1Buf2] for buffer [tt-1]
Note: 'force_dmo_alias' annotation [tt2] overrides remapping of [tt2Buf2] for buffer [tt2]
Note: 'force_dmo_alias' annotation [tt1] overrides remapping of [tt1Buf2] for buffer [tt-1]
Note: 'force_dmo_alias' annotation [tt3] overrides remapping of [tt3Buf2] for buffer [tt3]
Note: 'force_dmo_alias' annotation [tt4] overrides remapping of [tt4Buf2] for buffer [tt4]
Note: 'force_dmo_alias' annotation [tt5] overrides remapping of [tt5Buf2] for buffer [tt5]
Note: 'force_dmo_alias' annotation [tt1] overrides remapping of [tt1Buf2] for buffer [tt-1]
Note: 'force_dmo_alias' annotation [tt2] overrides remapping of [tt2Buf2] for buffer [tt2]
Note: 'force_dmo_alias' annotation [tt3] overrides remapping of [tt3Buf2] for buffer [tt3]
Note: 'force_dmo_alias' annotation [tt1] overrides remapping of [tt1Buf2] for buffer [tt-1]
Note: 'force_dmo_alias' annotation [tt2] overrides remapping of [tt2Buf2] for buffer [tt2]
Note: 'force_dmo_alias' annotation [tt4] overrides remapping of [tt4Buf2] for buffer [tt4]
Note: 'force_dmo_alias' annotation [tt5] overrides remapping of [tt5Buf2] for buffer [tt5]
Note: 'force_dmo_alias' annotation [tt1] overrides remapping of [tt1Buf2] for buffer [tt-1]
Note: 'force_dmo_alias' annotation [tt3] overrides remapping of [tt3Buf2] for buffer [tt3]
```

```
Note: 'force_dmo_alias' annotation [tt4] overrides remapping of [tt4Buf2] for buffer [tt4]
Note: 'force_dmo_alias' annotation [tt1] overrides remapping of [tt1Buf2] for buffer [tt-1]
Note: 'force_dmo_alias' annotation [tt2] overrides remapping of [tt2Buf2] for buffer [tt2]
Note: 'force_dmo_alias' annotation [tt3] overrides remapping of [tt3Buf2] for buffer [tt3]
Note: 'force_dmo_alias' annotation [tt5] overrides remapping of [tt5Buf2] for buffer [tt5]
Note: 'force_dmo_alias' annotation [tt3] overrides remapping of [tt3Buf2] for buffer [tt3]
Note: 'force_dmo_alias' annotation [tt4] overrides remapping of [tt4Buf2] for buffer [tt4]
```

in the server stdout.log.
Is it normal?
Thank you.

#23 - 04/17/2023 11:22 AM - Igor Skornyakov

Fixed issues with READ-JSON for empty dynamic DATA-SET revealed by additional tests.
Committed to 7193a/14543.

However these additional testes reveal a strange problem with WRITE-XML for the PARENT-ID-RELATION - in some situations the last parent record is not serialized. It looks strange because WRITE-JSON works fine and the iteration logic looks very similar in these methods.
Investigating.

#24 - 04/18/2023 04:47 AM - Igor Skornyakov

Igor Skornyakov wrote:

> However these additional testes reveal a strange problem with WRITE-XML for the PARENT-ID-RELATION - in some situations the last parent record is not serialized. It looks strange because WRITE-JSON works fine and the iteration logic looks very similar in these methods.

Actually, the problem is not with WRITE-XML *per se*, but with the **very first WRITE-XML/WRITE-JSON after READ-JSON**.
Analysing...

#25 - 04/18/2023 06:42 AM - Igor Skornyakov

Igor Skornyakov wrote:

> Igor Skornyakov wrote:
>
> > However these additional testes reveal a strange problem with WRITE-XML for the PARENT-ID-RELATION - in some situations the last parent record is not serialized. It looks strange because WRITE-JSON works fine and the iteration logic looks very similar in these methods.
>
> Actually, the problem is not with WRITE-XML *per se*, but with the **very first WRITE-XML/WRITE-JSON after READ-JSON**.
> Analysing...

Indeed, I see with a debugger that right after READ-JSON for the table which is not serialized completely with subsequent WRITE-JSON/WRITE-XML the content of the buffer is not seen in the buffer's tempTableRef.
What is the right way to 'flush' the buffer?
Thank you.

#26 - 04/18/2023 07:36 AM - Igor Skornyakov

Well, all tests passed.
Committed to 7193a/14544.

Please review.
Thank you.

#27 - 04/19/2023 02:20 PM - Vladimir Tsichevski

I am running OEUnit fixture test class:

```
&Scoped-define CUSTOMER1-NAME John Smith
&Scoped-define CUSTOMER1-NUM 10
&Scoped-define CUSTOMER2-NAME Freddie Mercury
&Scoped-define CUSTOMER2-NUM 11

USING OEUnit.Assertion.Assert.
USING OEUnit.Data.Fixture.

ROUTINE-LEVEL ON ERROR UNDO, THROW.

CLASS examples.FixtureJSON:

  @Test(fixture = "FixtureFromJSON").
  METHOD PUBLIC VOID testRunFixtureJSON():
    DEFINE VARIABLE recordCount AS INTEGER NO-UNDO.
    FOR EACH test3820.Customer:
      Assert:IsTrue(("{&CUSTOMER1-NAME}" = customerName AND {&CUSTOMER1-NUM} = customerNum)
                 OR ("{&CUSTOMER2-NAME}" = customerName AND {&CUSTOMER2-NUM} = customerNum)).
      recordCount = recordCount + 1.
    END.
    Assert:AreEqual(recordCount, 2).
  END METHOD.

  @Fixture(name = "FixtureFromJSON").
  METHOD PUBLIC Fixture myFixture_JSON():
    DEFINE VARIABLE fixture AS Fixture NO-UNDO.
    fixture = NEW Fixture().
    fixture:FromJSON("~{ ~"Customer~": ["
                     + "  ~{ ~"customerName~": ~"{&CUSTOMER1-NAME}~", ~"customerNum~": {&CUSTOMER1-NUM}},"
                     + "  ~{ ~"customerName~": ~"{&CUSTOMER2-NAME}~", ~"customerNum~": {&CUSTOMER2-NUM}},"
                     + "]}").
    Assert:AreEqual(fixture:TableCount, 2).
    Assert:AreEqual(fixture:Size, 1).
    RETURN fixture.
  END METHOD.

END CLASS.
```

In this class a fixture is reading its dataset from JSON code. The code is:

```
{ "Customer": [  { "customerName": "John Smith", "customerNum": 10},  { "customerName": "Freddie Mercury", "cu
stomerNum": 11}]}
```

the code reads nothing here, the very first token read is null:

```
      try (Reader reader = new InputStreamReader(source.getStream());
           JsonParser parser = factory.createParser(reader))
      {
         this.ds = ds;
         this.parser = parser;
         this.peerMapping = new HashMap<>();

         JsonToken tok = parser.nextToken();
         if (tok != START_OBJECT)
         {
            // TODO: proper message and error reporting
            return false;
         }
```

#28 - 04/19/2023 02:45 PM - Igor Skornyakov

Vladimir Tsichevski wrote:

> I am running OEUnit fixture test class:
>
> [...]
>
> In this class a fixture is reading its dataset from JSON code. The code is:
>
> [...]
>
> the code reads nothing here, the very first token read is null:
>
> [...]

The { "Customer": [{ "customerName": "John Smith", "customerNum": 10}, { "customerName": "Freddie Mercury", "customerNum": 11}]} does not look like DATA-SET. I've tested READ-JSON with sources which are result of WRITE-JSON. The result of WRITE-JSON always start with the { "${dsname}".
Can you please provide how the result of READ-JSON looks with 4GL?
Thank you.

BTW: Have you tested with 7193a? It is still on the code review and was not merged to the trunk yet.

#29 - 04/19/2023 03:36 PM - Ovidiu Maxiniuc

Review of 7193a/14544.

Good job! A lot of code with adjustments for existing features. Here are some notes, most of them related to code aspect:

- AbstractTempTable.java: please set the number in H entry (21);
- BufferImpl.java: please delete line 271 (unwanted copy of 266);
- RecordBuffer.java:
  - invalid H entry (should be 327, but probably will be changed again after rebase operation);
  - OTOH, loadRecord() was expressly made unavailable to public. As noted in javadoc, *this is a dedicated method accessible only from a very specific places within the persistence package*. Can we avoid direct call to it from other packages?
- JsonExport.java: in method exportDataSet(): I think the filtering of the two stream processing can be merged into a single construct (as in XmlExport.java:729). That is, unless the order of the filtered relations is important. Also, please indent/move the forEach together with . punctuation: it is difficult to read as a single statement, especially for someone used with 4GL syntax with DOT statement separator;
- JsonImport.java:
  - please move the H number on first history line (36) instead of last (39) of the new entry;
  - createTableStructure() I see you added a new return value (null) by switching the return type to Boolean. Please document the change in javadoc.
  - lines 505-509, 589-593, 1187-11491: please use the same indentation for all method parameters. The methods (readTable) are pretty similar, I wonder whether we can merge them in a single method?
  - relations streaming at lines 743-747, see similar note in JsonExport.java;
  - parseRecord method, exit-points at lines 1146 and 1152: are there specific errors which should be presented to user?
  - line 1741: please fix javadoc indentation. Add an empty line before @throws section;
  - LE: ErrorManager.java: please put/move your new logs in H section under a new entry at the end, after the last entry already existing in trunk.

Note that at the moment of review the branch was based on trunk/14536. It needs to be rebased before merging into trunk.

#30 - 04/19/2023 03:48 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Review of 7193a/14544.
>
> Good job! A lot of code with adjustments for existing features. Here are some notes, most of them related to code aspect:
>
> - AbstractTempTable.java: please set the number in H entry (21);
> - BufferImpl.java: please delete line 271 (unwanted copy of 266);
> - RecordBuffer.java:
>   - invalid H entry (should be 327, but probably will be changed again after rebase operation);
>   - OTOH, loadRecord() was expressly made unavailable to public. As noted in javadoc, *this is a dedicated method accessible only from a very specific places within the persistence package.* Can we avoid direct call to it from other packages?
> - JsonExport.java: in method exportDataSet(): I think the filtering of the two stream processing can be merged into a single construct (as in XmlExport.java:729). That is, unless the order of the filtered relations is important. Also, please indent/move the forEach together with . punctuation: it is difficult to read as a single statement, especially for someone used with 4GL syntax with DOT statement separator;
> - JsonImport.java:
>   - please move the H number on first history line (36) instead of last (39) of the new entry;
>   - createTableStructure() I see you added a new return value (null) by switching the return type to Boolean. Please document the change in javadoc.

- lines 505-509, 589-593, 1187-11491: please use the same indentation for all method parameters. The methods (readTable) are pretty similar, I wonder whether we can merge them in a single method?
- relations streaming at lines 743-747, see similar note in JsonExport.java;
- parseRecord method, exit-points at lines 1146 and 1152: are there specific errors which should be presented to user?
- line 1741: please fix javadoc indentation. Add an empty line before @throws section;
- LE: ErrorManager.java: please put/move your new logs in H section under a new entry at the end, after the last entry already existing in trunk.

Note that at the moment of review the branch was based on trunk/14536. It needs to be rebased before merging into trunk.

Thank you, Ovidiu.
I will address it tomorrow.

#31 - 04/19/2023 03:53 PM - Vladimir Tsichevski

- File TestDataset.cls added

Igor Skornyakov wrote:

> The { "Customer": [{ "customerName": "John Smith", "customerNum": 10}, { "customerName": "Freddie Mercury", "customerNum": 11}]} does not look like DATA-SET. I've tested READ-JSON with sources which are result of WRITE-JSON. The result of WRITE-JSON always start with the { "${dsname}".
> Can you please provide how the result of READ-JSON looks with 4GL?
> Thank you.

I've attached an ABLUnit test class for dataset import. All tests in this class succeed in OE.

> BTW: Have you tested with 7193a? It is still on the code review and was not merged to the trunk yet.

No, I've extracted the diff applied to my local environment, which is rebased to the latest trunk.

But from what you've said, you do not anticipate the JSON structure I've provided, so the problem is not in different environments.

#32 - 04/19/2023 04:05 PM - Igor Skornyakov

Vladimir Tsichevski wrote:

> Igor Skornyakov wrote:
>
> > The { "Customer": [{ "customerName": "John Smith", "customerNum": 10}, { "customerName": "Freddie Mercury", "customerNum": 11}]} does not look like DATA-SET. I've tested READ-JSON with sources which are result of WRITE-JSON. The result of WRITE-JSON always start with the { "${dsname}".
> > Can you please provide how the result of READ-JSON looks with 4GL?
> > Thank you.
>
> I've attached an ABLUnit test class for dataset import. All tests in this class succeed in OE.
>
> > BTW: Have you tested with 7193a? It is still on the code review and was not merged to the trunk yet.

As far as I understand the 7193a is not reabsed to lasterst trunk.

> No, I've extracted the diff applied to my local environment, which is rebased to the latest trunk.

As far as I understand the 7193a is not reabsed to lasterst trunk. But I do not think that it is matters.

> But from what you've said, you do not anticipate the JSON structure I've provided, so the problem is not in different environments.

If you believe that you test case should be exactly like it is now, I will need to add support for your test case.
Where can I find a desription how to run your test?
Thank you.

#33 - 04/19/2023 04:13 PM - Vladimir Tsichevski

Igor Skornyakov wrote:

> As far as I understand the 7193a is not reabsed to lasterst trunk.
>
> > No, I've extracted the diff applied to my local environment, which is rebased to the latest trunk.
>
> As far as I understand the 7193a is not reabsed to lasterst trunk. But I do not think that it is matters.

Yes and yes.

> But from what you've said, you do not anticipate the JSON structure I've provided, so the problem is not in different environments.
>
> If you believe that you test case should be exactly like it is now, I will need to add support for your test case.
> Where can I find a desription how to run your test?

At my brand new page :-) here:
https://proj.goldencode.com/projects/p2j/wiki/4GL_Unit_Testing

Please, feel free to ask questions about ABLUnit! The thing is just released, so some aspects of configuration could be missing or unclear in the docs.

#34 - 04/20/2023 05:12 AM - Igor Skornyakov

The task branch 7193a was rebased to the trunk/14543.
Pushed up to revision 14551.

#35 - 04/20/2023 06:48 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

Review of 7193a/14544.

- AbstractTempTable.java: please set the number in H entry (21);

Done.

- BufferImpl.java: please delete line 271 (unwanted copy of 266);

Done.

- RecordBuffer.java:
  - invalid H entry (should be 327, but probably will be changed again after rebase operation);
  - OTOH, loadRecord() was expressly made unavailable to public. As noted in javadoc, *this is a dedicated method accessible only from a very specific places within the persistence package*. Can we avoid direct call to it from other packages?

H entry fixed, refactored code to use loadRecord() from the TaempTable Builder.

- JsonExport.java: in method exportDataSet(): I think the filtering of the two stream processing can be merged into a single construct (as in XmlExport.java:729). That is, unless the order of the filtered relations is important. Also, please indent/move the forEach together with . punctuation: it is difficult to read as a single statement, especially for someone used with 4GL syntax with DOT statement separator;

Done.

- JsonImport.java:
  - please move the H number on first history line (36) instead of last (39) of the new entry;

Done.

- createTableStructure() I see you added a new return value (null) by switching the return type to Boolean. Please document the change in javadoc.

Done.

- lines 505-509, 589-593, 1187-11491: please use the same indentation for all method parameters. The methods (readTable) are pretty similar, I wonder whether we can merge them in a single method?

Done.

- relations streaming at lines 743-747, see similar note in JsonExport.java;

In this case we consider only FOREIGN-KEY-HIDDEN relations since for PARENT-ID-RELATION the logic is different.

- parseRecord method, exit-points at lines 1146 and 1152: are there specific errors which should be presented to user

I'm not sure. It looks that these exits should 'never happen' unless the source JSON is badly broken.

- line 1741: please fix javadoc indentation. Add an empty line before @throws section;

Done.

- LE: ErrorManager.java: please put/move your new logs in H section under a new entry at the end, after the last entry already existing in trunk.

Done.

Note that at the moment of review the branch was based on trunk/14536. It needs to be rebased before merging into trunk.

The branch was rebased to the most recent version of the trunk.

Committed to 7193a/14552.

Working on support og the Vladimir's test case (#7193-27).

#36 - 04/20/2023 01:55 PM - Igor Skornyakov

I've created a more complete test for inferring empty dynamic DATA-SET from JSON data w/o root element (#7193-27).
Now FWD correctly supports this.
Committed to 7193/14553.

In this situation a DATA-SET with NAME = "NewDataSet" and SERIALIZE-HIDDEN = true is created.
I've noticed however that FWD does not support DATA-SET:SERIALIZE-HIDDEN in WRITE-XML/WRITE-XMLSCHEMA.
Working on this.

#37 - 04/28/2023 04:11 PM - Igor Skornyakov

- File json1.p added

- File xds.xml added

- File xds.xsd added

Implemented DATA-SET:WRITE-XMLSCHEMA with SERIALIZE-HIDDEN = true.
Committed to 7193a/14566.

This provides the same result as 4GL **if the order of tables in the DATA-SET is compatible with parent/child partial ordering**.

Please review.

Please note that if it is not the fact, 4GL WRITE-XML output doesn't look correct. See for example attached test and 4GL output files.

Another problem is with situation when the DATA-SET with SERIALIZE-HIDDEN = true contains multiple namespaces. In this case the 4GL WRITE-XML output is just not well-formed.

If you believe that we have to reproduce the quirks mentioned above I suggest to deeal with it in sce scope of #7247.
What do you think?
Thank you.

#38 - 04/29/2023 06:05 AM - Igor Skornyakov

Igor Skornyakov wrote:

> Implemented DATA-SET:WRITE-XMLSCHEMA with SERIALIZE-HIDDEN = true.
> Committed to 7193a/14566.

Sorry, I've noticed some issues with 7193a/14566 that must be fixed.

#39 - 05/01/2023 06:33 AM - Greg Shah

> If you believe that we have to reproduce the quirks mentioned above I suggest to deeal with it in sce scope of #7247.

Ovidiu: Please assess if these quirks must be fixed or if they can be deferred.

#40 - 05/02/2023 02:07 PM - Igor Skornyakov

Finished support for DATA-SET:SERIALIZE-HIDDEN.
Committed to 7193a/14567.

Now FWD provides exactly the same output for WRITE-XML for DATA-SET with SERIALIZE-HIDDEN = true **if the dataset satisfies the following conditions**:

1. It has at most one XML namespece
2. The order of tables in the dataset is compatible with the parent/child tables' ordering.

This includes the situation when the first table of the dataset has at least one field with XML-NODE-TYPE 'attribute'. In this case 4GL reports the warnings 13065, 13093 and produces an incomplete output.

If the conditions mentioned above are not met, 4GL typically produces wierd or even not well-formed output (see #7193-37). I suggest to deal with this not in the scope of this task (if it makes sense at all).

Please review my changes.
Thank you.

#41 - 05/02/2023 02:09 PM - Greg Shah

Ovidiu: Please review.

#42 - 05/02/2023 04:57 PM - Igor Skornyakov

4GL and FWD format ouput JSON files in not exactly the same way. To simplify comparision of 4GL and FWD results I use jq (https://www.baeldung.com/linux/jq-command-json) and json-glib-format (https://developer-old.gnome.org/json-glib/unstable/json-format-tool.html) utilities. The second one is required because jq fails to process 4GL output for some reason I had no time to investigate.
I've defined the followin aliases:

```
alias _JSF='_jsf(){ json-glib-format $1 | jq -c | jq -M >x.json; mv x.json $1;}; _jsf'
alias JSF='jsf(){ find $1 -name "*.json" | while read f; do _JSF $f; done;}; jsf'
```

Now the command

```
JSF <target>
```

re-formats all *.json files in the <target> directory and we can compare e.g. 4GL and FWD output.

Maybe this can be useful.

#43 - 05/03/2023 07:50 PM - Ovidiu Maxiniuc

Review of 7193a/14567, (re-)based on trunk 14552:

- TemporaryBuffer.java: The H note mentions "processed *errors*" but I think you meant the "processed *rows*", as in javadoc @return tag.
- JsonImport.java:
  - there is a couple of IllegalStateException thrown (lines 1251 and 1412). Looking upstream the call flow, there is no try catching them. As result they will propagate up to converted code causing the application to abend. This is not the correct handling of the exceptional situation you identified. Please replace these with the appropriate legacy errors 4GL raise in these cases;
  - line 1409. I see you like the collection streaming API. Indeed, it maybe more concise, using less vertical space. However, although not banned, these constructs are not encouraged in GCD mostly because of the overhead added by their execution;
  - line 1771/1772: please reformat the javadoc code;
  - line 1773: push throws on next line;
  - line 1817: a munch faster way to rewrite it is: ttb = (TempTableBuilder) tt.getResource();. Unwrapping is used in converted code or when there is a chance that the value store in handle to be incorrect. In this case things are clear, the dynamic temp-table was created the line above;
  - the new method readNested(). reading the code I understand it always returns one of two values: null on fail - as described in javadoc and END_OBJECT on success. That is because the invariant of the last do/while block of the method only allows to exit the loop with this value for tok which is then returned.
  - please separate the @param from method description and @return / @throw in javadoc for methods readTable() persist(), typeOf(), readNested(), setParentValues();
  - you created the tt*** Stack s for storing different information. They are always in synch thanks to methods pushContext() and popContext().

I suspect that a solution with better performance would be to group all these 6 pieces of information into a single context structure and store it in a single Stack instead.

- XmlExport.java
  - I like the work done here. The result is more structured and xml formatting happens more fluent.
  - typos at line 462, 499: Fiaild / Faild
  - as for source above, use an empty line between @ sections in javadoc.
  - the Fatal exception. It is thrown (indirectly) when attrFailure() in invoked (after some 4GL error raised and in silent mode). This runtime exception is only caught from methods calls initiated from DataSet, but not for simple tables. The the latter case the exception cases will lead to client to abend.

#44 - 05/04/2023 04:41 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Review of 7193a/14567, (re-)based on trunk 14552:
>
> - TemporaryBuffer.java: The H note mentions "processed *errors*" but I think you meant the "processed *rows*", as in javadoc @return tag.

Fixed.

- JsonImport.java:
  - there is a couple of IllegalStateException thrown (lines 1251 and 1412). Looking upstream the call flow, there is no try catching them. As result they will propagate up to converted code causing the application to abend. This is not the correct handling of the exceptional situation you identified. Please replace these with the appropriate legacy errors 4GL raise in these cases;

These situations 'should never happen' - it is just a precaution if the source JSON is badly broken. I've replaced IllegalStateException with PersistenceException.

- line 1409. I see you like the collection streaming API. Indeed, it maybe more concise, using less vertical space. However, although not banned, these constructs are not encouraged in GCD mostly because of the overhead added by their execution;

I do not think that JsonImport is called very often in real apps, so the overhead is not that important comparing to the code clarity and maintainability.

- line 1771/1772: please reformat the javadoc code;

Done.

- line 1773: push throws on next line;

Done.

- line 1817: a munch faster way to rewrite it is: ttb = (TempTableBuilder) tt.getResource();. Unwrapping is used in converted code or when there is a chance that the value store in handle to be incorrect. In this case things are clear, the dynamic temp-table was created the line above;

Fixed.

- the new method readNested(). reading the code I understand it always returns one of two values: null on fail - as described in javadoc and END_OBJECT on success. That is because the invariant of the last do/while block of the method only allows to exit the loop with this value

for tok which is then returned.

AFAIR this method had different versions, and in some of them it returned different values. I prefer to retain more generic description since logically is is more important that this method returns last seen JsonToken rather than exact type of this token.

- please separate the @param from method description and @return / @throw in javadoc for methods readTable() persist(), typeOf(), readNested(), setParentValues();

Done.

- you created the tt*** Stack s for storing different information. They are always in synch thanks to methods pushContext() and popContext(). I suspect that a solution with better performance would be to group all these 6 pieces of information into a single context structure and store it in a single Stack instead.

I totally agree. These stacks where added one by one. Grouping them in a single context will require a lot (albeit trivial) changes. The performance improvement seems questionable since this refactoring will add indirection on access to the context members. This is why I finally decided to leave it as-is.

- XmlExport.java
  - I like the work done here. The result is more structured and xml formatting happens more fluent.

Thank you.

- typos at line 462, 499: Fiaild / Faild

Fixed.

- as for source above, use an empty line between @ sections in javadoc.

Fixed (where noticed).

- the Fatal exception. It is thrown (indirectly) when attrFailure() in invoked (after some 4GL error raised and in silent mode). This runtime exception is only caught from methods calls initiated from DataSet, but not for simple tables. The the latter case the exception cases will lead to client to abend.

attrFailure() is called only when serializeHidden is true which can be only for DataSet. However, I've added the corresponding catch to the BufferImpl.writeXml just in case.

Committed to 7193a/14568.

Can I merge it to the trunk?
Thank you.

#45 - 05/04/2023 07:50 AM - Greg Shah

> I do not think that JsonImport is called very often in real apps

JSON processing will be very heavily used in REST applications. We have customer applications that are completely based on REST APIs. If this is even used once per API call, then it is in fact heavily used since it will be called millions of times per day. Perhaps I'm not thinking about this correctly, but I'd need more of a reason to accept that it is very rarely used.

With that in mind, the streams should be removed.

> > you created the tt*** Stack s for storing different information. They are always in synch thanks to methods pushContext() and popContext(). I suspect that a solution with better performance would be to group all these 6 pieces of information into a single context structure and store it in a single Stack instead.
>
> I totally agree. These stacks where added one by one. Grouping them in a single context will require a lot (albeit trivial) changes. The performance improvement seems questionable since this refactoring will add indirection on access to the context members. This is why I finally decided to leave it as-is.

We've found that runtime code measurably slows down when one uses 5 stacks (or any large number) instead of 1 for the same basic scoping feature. There is a real cost to pushing/popping so many stacks since there is non-trivial processing involved, including the creation/copying/removal of internal data structures. An additional level of dereferencing is a small price to pay for the improvement in performance.

Please resolve this now and we will take any small performance improvement it offers.

#46 - 05/04/2023 10:26 AM - Igor Skornyakov

Greg Shah wrote:

> We've found that runtime code measurably slows down when one uses 5 stacks (or any large number) instead of 1 for the same basic scoping feature. There is a real cost to pushing/popping so many stacks since there is non-trivial processing involved, including the creation/copying/removal of internal data structures. An additional level of dereferencing is a small price to pay for the improvement in performance.

> Please resolve this now and we will take any small performance improvement it offers.

Started to work on this. Strangely it appears to be not that strigntforward as I expected.
Still working.

#47 - 05/04/2023 10:41 AM - Igor Skornyakov

Igor Skornyakov wrote:

> Greg Shah wrote:
>
> > We've found that runtime code measurably slows down when one uses 5 stacks (or any large number) instead of 1 for the same basic scoping feature. There is a real cost to pushing/popping so many stacks since there is non-trivial processing involved, including the creation/copying/removal of internal data structures. An additional level of dereferencing is a small price to pay for the improvement in performance.
> >
> > Please resolve this now and we will take any small performance improvement it offers.
>
> Started to work on this. Strangely it appears to be not that strigntforward as I expected.
> Still working.

Refactored to use a single stack.
Committed to 7193a/14569.

#48 - 05/04/2023 02:33 PM - Greg Shah

Please remove the streams usage.

#49 - 05/04/2023 02:43 PM - Igor Skornyakov

Greg Shah wrote:

> Please remove the streams usage.

Done.

Committed to 7193a/14570.

#50 - 05/04/2023 02:43 PM - Greg Shah

Ovidiu: Please review.

#51 - 05/04/2023 02:44 PM - Greg Shah

Igor: Please set the % Done.

#52 - 05/04/2023 02:45 PM - Igor Skornyakov

- % Done changed from 0 to 100

#53 - 05/04/2023 10:28 PM - Ovidiu Maxiniuc

Review of 7193a/14570 (compared to previous iteration, 14567).

- JsonImport.java
  - line 914: you replaced relations.stream().forEach() with for(Pair<String, String> p: relations). Knowing that relations is an ArrayList<>, a faster method is to iterate it without creating an iterator. That is, an indexed for loop, and access the elements of the list using get(i);
  - lines 1794/1834: using try/finally is the right way for keeping the context stack. I failed to see this in my previous review. With a single stack now, the pushContext() and popContext() can even be inlined.

Please rebase and redo your local tests if you have not done since r14567.

Greg,
Before merging this branch to trunk we should test it against Z/POC application. I can do it after the rebase. There are some tests in xfer/testcases project I guess, but I do not think I have time for them.

#54 - 05/05/2023 02:57 AM - Igor Skornyakov

Rebased 7193a from the trunk/14560.

Pushed up to revision 14578.

Re-testing.

#55 - 05/05/2023 05:25 AM - Greg Shah

> Before merging this branch to trunk we should test it against Z/POC application.

Ovidiu: Yes, please do.

#56 - 05/05/2023 05:35 AM - Igor Skornyakov

Greg Shah wrote:

> Before merging this branch to trunk we should test it against Z/POC application.

Ovidiu: Yes, please do.

At this moment I'm running my old tests and found a number of regressions. Working on them now.
Maybe it makes sense to run Ovidiu's tests after I will finish my testing?
Thank you.

#57 - 05/05/2023 05:48 AM - Greg Shah

Agreed.

#58 - 05/05/2023 05:48 AM - Greg Shah

- Status changed from Review to WIP

#59 - 05/06/2023 01:49 PM - Igor Skornyakov

- Status changed from WIP to Review

Ovidiu Maxiniuc wrote:

> Review of 7193a/14570 (compared to previous iteration, 14567).
>
> - JsonImport.java
>   - line 914: you replaced relations.stream().forEach() with for(Pair<String, String> p: relations). Knowing that relations is an ArrayList<>, a faster method is to iterate it without creating an iterator. That is, an indexed for loop, and access the elements of the list using get(i);

Done.

- lines 1794/1834: using try/finally is the right way for keeping the context stack. I failed to see this in my previous review. With a single stack now, the pushContext() and popContext() can even be inlined.

Actually pushContext() is not a sinle liner and is called from at least two places. So I decided to leave it as-as.

> Please rebase and redo your local tests if you have not done since r14567.

I've rebased 71993a to the trunk/14560.
When running my test suite for DATASET/TEMP-TABLE (de)serialization a number of regessions where found and fixed.

The final (I hope) version is committed to 7193a/14579.

#60 - 05/06/2023 01:53 PM - Igor Skornyakov

Recently I started to see the following exception at the end of the test session close:

```
[05/06/2023 20:43:02 GMT+03:00] (com.goldencode.p2j.net.HighLevelObject:SEVERE) Failed to obtain routing key
java.lang.IllegalStateException: The queue is not running!
        at com.goldencode.p2j.net.Queue.checkState(Queue.java:1354)
        at com.goldencode.p2j.net.Queue.transact(Queue.java:671)
        at com.goldencode.p2j.net.BaseSession.transact(BaseSession.java:271)
        at com.goldencode.p2j.net.HighLevelObject.transact(HighLevelObject.java:220)
        at com.goldencode.p2j.net.HighLevelObject.getKey(HighLevelObject.java:172)
        at com.goldencode.p2j.net.RemoteObject$RemoteAccess.obtainRoutingKey(RemoteObject.java:1551)
        at com.goldencode.p2j.net.RemoteObject$RemoteAccess.invokeCore(RemoteObject.java:1463)
        at com.goldencode.p2j.net.InvocationStub.invoke(InvocationStub.java:144)
        at com.sun.proxy.$Proxy20.cleanup(Unknown Source)
        at com.goldencode.p2j.util.memptr$1.cleanup(memptr.java:211)
        at com.goldencode.p2j.util.memptr$1.cleanup(memptr.java:189)
        at com.goldencode.p2j.security.ContextLocal$Wrapper.cleanup(ContextLocal.java:822)
        at com.goldencode.p2j.security.SecurityContext.reset(SecurityContext.java:692)
        at com.goldencode.p2j.security.SecurityContext.cleanupWorker(SecurityContext.java:671)
        at com.goldencode.p2j.security.SecurityContext.cleanupWorker(SecurityContext.java:596)
        at com.goldencode.p2j.security.SecurityContext.cleanup(SecurityContext.java:574)
        at com.goldencode.p2j.security.SecurityManager.terminateSessionWorker(SecurityManager.java:4676)
        at com.goldencode.p2j.security.SecurityManager.terminateSession(SecurityManager.java:4579)
        at com.goldencode.p2j.net.BaseSession.cleanupContext(BaseSession.java:415)
        at com.goldencode.p2j.net.BaseSession.end(BaseSession.java:463)
        at com.goldencode.p2j.net.SessionManager.endSession(SessionManager.java:1380)
        at com.goldencode.p2j.net.SessionManager.queueStopped(SessionManager.java:1821)
        at com.goldencode.p2j.net.Queue.stop(Queue.java:503)
        at com.goldencode.p2j.net.Protocol.stopQueue(Protocol.java:424)
        at com.goldencode.p2j.net.Protocol.access$700(Protocol.java:166)
        at com.goldencode.p2j.net.Protocol$Reader.run(Protocol.java:536)
        at java.lang.Thread.run(Thread.java:750)
```

This happens only for tests that use memptr.

#61 - 05/08/2023 01:55 PM - Greg Shah

> Recently I started to see the following exception at the end of the test session close:
> [...]
>
> This happens only for tests that use memptr.

Please try to isolate a standalone recreate.

#62 - 05/08/2023 02:14 PM - Igor Skornyakov

- File mptr.p added

Greg Shah wrote:

> > Recently I started to see the following exception at the end of the test session close:
> > [...]
> >
> > This happens only for tests that use memptr.
>
> Please try to isolate a standalone recreate.

Please see the attached 'almost minimal' test.

#63 - 05/08/2023 02:29 PM - Greg Shah

- Related to Bug #4071: FWD server stops allowing new web client logins after abend added

#64 - 05/10/2023 10:04 PM - Ovidiu Maxiniuc

I reviewed 7193a/14580 and found no problems in this last commit.

I understand that the issue from note #7193-60 is irrelevant to this task (the memptr deallocation attempt after the connection to client is interrupted/suspended) so it can be ignored here.

The test against a large application found no regression. I do not have the ETF environment prepared so I cannot do that test, but probably the test is meaningless because this task handles newer features.

#65 - 05/11/2023 02:24 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I reviewed 7193a/14580 and found no problems in this last commit.
>
> I understand that the issue from note #7193-60 is irrelevant to this task (the memptr deallocation attempt after the connection to client is interrupted/suspended) so it can be ignored here.
>
> The test against a large application found no regression. I do not have the ETF environment prepared so I cannot do that test, but probably the test is meaningless because this task handles newer features.

Thank you, Ovidiu.

Eric, Greg - can I merge the task branch to the trunk?
Thank you.

#66 - 05/11/2023 05:35 AM - Greg Shah

> I understand that the issue from note #7193-60 is irrelevant to this task (the memptr deallocation attempt after the connection to client is interrupted/suspended) so it can be ignored here.

Igor: Please create a new task for this and post the recreate, link it here as a Related Task.

> can I merge the task branch to the trunk?

Yes

#67 - 05/11/2023 06:05 AM - Igor Skornyakov

Greg Shah wrote:

> can I merge the task branch to the trunk?

> Yes

The branch 7123a was merged to the trunk rev. 14566 and archived.

#68 - 05/11/2023 06:08 AM - Igor Skornyakov

- Related to Bug #7345: The memptr deallocation attempt after the connection to client is interrupted/suspended added

#69 - 05/11/2023 06:09 AM - Igor Skornyakov

Greg Shah wrote:

> Igor: Please create a new task for this and post the recreate, link it here as a Related Task.

Done. See #7345

#70 - 05/26/2023 02:09 PM - Eric Faulhaber

- Related to Bug #6475: double-check and fix any read/write-xml/json issues at table and dataset added

#71 - 06/12/2023 02:20 PM - Greg Shah

- Status changed from Review to Closed

Files

json.p	8.94 KB	03/14/2023	Igor Skornyakov
json-fk.p	13.1 KB	03/26/2023	Igor Skornyakov
json.p	13 KB	03/26/2023	Igor Skornyakov
ds.1.json	2.21 KB	03/26/2023	Igor Skornyakov
ds.1.fk.json	1.15 KB	03/26/2023	Igor Skornyakov
ds.json	2.21 KB	03/26/2023	Igor Skornyakov
json-fk.p	14 KB	04/17/2023	Igor Skornyakov
TestDataset.cls	2.43 KB	04/19/2023	Vladimir Tsichevski
json1.p	14.5 KB	04/28/2023	Igor Skornyakov
xds.xml	9.03 KB	04/28/2023	Igor Skornyakov
xds.xsd	15.4 KB	04/28/2023	Igor Skornyakov
mptr.p	3.02 KB	05/08/2023	Igor Skornyakov