Base Language - Bug #4749

Conversion error for object indeterminate extent output parameter

07/08/2020 03:57 AM - Marian Edu

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

Description

History

#1 - 07/29/2020 06:06 AM - Marian Edu

- Assignee set to Greg Shah

Assigning that to Greg to dispatch it to the right recipient :)

When we have a method that returns an object array (indeterminate extent) the conversion uses a cast to assign back the result of assignMulti, the problem is that the cast is not done to the converted java class as it should but to the 4GL class name which makes the converted code invalid. I think the output-parameter.rules is not finding the full-java-class annotation at that point.

We have the issue in oo/openedge/net/multipart_entity/test_method_get_part_character.p but I've also wrote a small testcase to isolate the issue:
oo/quirks/method_return_extent/ExtentMethod.cls
oo/quirks/method_return_extent/test_method_return_extent.oo.p

#2 - 07/30/2020 09:33 AM - Greg Shah

- Description updated

- Assignee changed from Greg Shah to Constantin Asofiei

- Start date deleted (07/08/2020)

#3 - 08/03/2020 05:26 PM - Constantin Asofiei

- Status changed from New to WIP

I have a fix for this, but is part of the #4824 work. Will commit with that.

#4 - 08/04/2020 12:08 PM - Constantin Asofiei

- % Done changed from 0 to 100

Fixed in 3821c rev 11436.

#5 - 08/04/2020 02:11 PM - Greg Shah

- Status changed from WIP to Test

#6 - 12/02/2020 07:04 AM - Marian Edu

To follow-up, the conversion issue is fixed indeed but now we have an issue with assignMulti when indeterminate array variables are used. This fails mainly because the indeterminate array is not seen as 'dynamic' and then FWD is throwing an error stating the size of the two arrays doesn't match.

I've tried to fix that by adding an ArrayAssigner.registerDynamicArray on TypeFactory.objectExtent which does fix that error but then if the variable is defined in procedure's main block doing that registration fails with NPE because these seems to be no scope active just yet.

Tried to make a few samples in oo/quirks/indeterminate_extent, only tried a couple of cases for now and the conversion seems to be different for 'global' variable compared to a local one in test_object_extent_method.p. Without registration of dynamic array both assignMulti fails, adding registration when the object extent is instantiated in TypeFactory the test will crash with NPE when global variable is initialized. This is only when we ran that code as a startup procedure, if we use a wrapper that in turns call RUN on it the the NPE is gone - there is already a valid `work area` scope.

The code generated for static property defined as object extent seems to be wrong as well - ObjectExtentGlobal.cls is not compiling in Java:

```
@LegacySignature(type = Type.VARIABLE, name = "goExt")
   private static ContextLocal<object<? extends com.goldencode.p2j.oo.lang._BaseObject_>[]> goExt = new Contex
tLocal<object<? extends com.goldencode.p2j.oo.lang._BaseObject_>[]>()
   {
      protected object<? extends com.goldencode.p2j.oo.lang._BaseObject_>[] initialValue()
      {
         return UndoableFactory.objectExtent(com.goldencode.p2j.oo.lang._BaseObject_.class);
      }
   };

...

goExt = ArrayAssigner.resize(goExt.get(), new integer());
goExt = ArrayAssigner.resize(goExt.get(), num);
```

#7 - 02/18/2021 10:22 AM - Greg Shah

- % Done changed from 100 to 80

- Status changed from Test to WIP

#8 - 02/19/2021 01:55 PM - Constantin Asofiei

The conversion issue for ObjectExtentGlobal is already fixed.

As for the dynamic array issues - the root cause was the fact that the Java reference for the parameter is changed by resize, and this reference must be updated in the OutputExtentParameter instance (as this needs to be set as an argument at the setVariable call, to replace the caller's reference).

#9 - 02/23/2021 12:45 PM - Constantin Asofiei

- % Done changed from 80 to 100

Constantin Asofiei wrote:

> As for the dynamic array issues - the root cause was the fact that the Java reference for the parameter is changed by resize, and this reference must be updated in the OutputExtentParameter instance (as this needs to be set as an argument at the setVariable call, to replace the caller's reference).

The fix is in 3821c 12052.

#10 - 02/24/2021 08:40 AM - Greg Shah

- Status changed from WIP to Test

Code Review Task Branch 3821c Revision 12052

Overall, the changes are good.

Please add javadoc to ArrayAssigner.register(BaseDataType[] array) and ArrayAssigner.registerDynamicArray(BaseDataType[] array) to make it clear the default value for pending. This is especially important since the default is different between these.

#11 - 03/06/2021 08:01 AM - Constantin Asofiei

Greg Shah wrote:

> Please add javadoc to ArrayAssigner.register(BaseDataType[] array) and ArrayAssigner.registerDynamicArray(BaseDataType[] array) to make it clear the default value for pending. This is especially important since the default is different between these.

Done in 3821c rev 12087.

#12 - 03/06/2021 10:17 AM - Greg Shah

Code Review Task Branch 3821c Revision 12087

The changes are good.