Base Language - Bug #3183

recursive use of some resources is sometimes broken

09/20/2016 06:48 PM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

Related issues:

Related to Conversion Tools - Feature #1735: leverage lambda expressions (Jav... **WIP**

History

#1 - 09/20/2016 07:10 PM - Greg Shah

Some issues with recursion have been found in #1735-50.

Constantin created 3 testcases. I have saved them into testcases/uast/. This task is meant to explore the extent of the problem (more than just these testcases) and to fix the deviations from the 4GL.

1. testcases/uast/recursive_use_of_queries_and_frames.p

This case has multiple problems.

- There is a use of a CompoundQuery that causes the query to be promoted to an instance member. The subsequent recursive usage of this query is broken.
- Frames are always promoted to instance members. This shows how recursion can be broken for frames.
- We are mis-handling the buffer and query references. Where the 4GL prints a simple numeric id for each handle, we fall-back to Object.toString() which definitely is wrong.

The 4GL output:

```
yes 1000 1001 1002
no 1003 1004 1005
no 1003 1004 1005
yes 1000 1001 1002
```

P2J output (trunk 11095 and also 1735e rev 11123):

```
yes com.goldencode.p2j.persist.$__Proxy2@44356f03 7204531883555451641
com.goldencode.p2j.persist.QueryWrapper@7d7aa7a0
no com.goldencode.p2j.persist.$__Proxy2@70f92bf7 6760024188750633638
com.goldencode.p2j.persist.QueryWrapper@7d032eb3
no com.goldencode.p2j.persist.$__Proxy2@70f92bf7 6760024188750633638
com.goldencode.p2j.persist.QueryWrapper@7d032eb3
yes com.goldencode.p2j.persist.$__Proxy2@44356f03 6760024188750633638
com.goldencode.p2j.persist.QueryWrapper@7d032eb3
```

2. testcases/uast/recursive_use_of_variables.p shows deviations since we are promoting the variable to be an instance member (so that it can be accessed in the trigger). Subsequent work in 1735e may fix this by making trigger inner classes and or lambdas contained inside of the internal proc (or function) so that the variables don't have to be promoted. There is at least 1 case where lambdas cannot be used but with this approach even the inner class case would work.

4GL output:

```
1 yes
1 no
1 no
1 yes
```

P2J output (trunk 11095):

```
1 yes
2 no
2 no
2 yes
```

3. testcases/uast/recursive_use_of_local_query.p is working the same in P2J as it does in the 4GL. There is nothing to do here except to confirm it still works.

Both generate this output:

```
10
20
20
```

4. We need to check other resource types. Consider what conditions can cause promotion for each resource and then try to create tests that exploit that.

5. We need to check functions and triggers.

#2 - 09/20/2016 07:12 PM - Greg Shah

- Related to Feature #1735: leverage lambda expressions (Java 8 "closures") to reduce code bloat added