Base Language - Bug #2375

fix the memptr initialization (in extent) and memptr parameter processing

08/13/2014 04:32 AM - Constantin Asofiei

Status:	Closed	**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	**version:**	

Description

Related issues:

Related to Base Language - Feature #1635: implement MEMPTR/RAW support	**Closed**	**01/19/2013**	**05/03/2013**
Related to Base Language - Bug #5608: raw vs memptr parameters	**New**		

History

#1 - 08/13/2014 04:47 AM - Constantin Asofiei

- File ca_upd20140813a.zip added

Attached update is a first pass at fixing the problems noted in #1635 note 29,30,31,32. Testcases which cover memptr used as arguments or in extent vars are found in testcases/uast/memptr/memptr_params* - use the memptrs_params_run.p to run the entire suite at once.

Known problems:

1. extent input-output memptr params are not initialized properly (see cases 0, 0b, 1, 1b)
2. input memptr params seem to alter the received argument (see cases 3, 3b, 4, 4b, 5, 5b, 6c).

#3 - 07/29/2021 09:36 AM - Constantin Asofiei

- Assignee set to Constantin Asofiei

- Status changed from New to WIP

#4 - 08/18/2021 05:33 AM - Constantin Asofiei

The 'memptr is both unknown and not-unknown' seems to not happen in Windows OE 11.6. Do you recall if the #1635-31 was tested with 4GL v9 under Linux?

I ask because the #1635-31 test in OE 11.6 outputs 'yes no yes no yes no' for the second line (the parameters checks for 'ne') and FWD was fixed to behave like this. I had to change some memptr/memptr_params* tests to pass in OE 11.6, and I don't recall what 4GL version was used to encode the expected errors/values.

I have fixed all known issues from my tests, as ran with OE 11.6. The solution was to group the (byteOrder, addr, size) fields from memptr class in a 'pointer' structure which is shared between the original argument and the memptr parameter. IT looks like 4GL passes the entire structure (i.e. memptr argument reference) to the parameter, but this is not the case in some unknown-assignment cases.

I'll clean up and commit the changes.

#5 - 08/18/2021 06:19 AM - Constantin Asofiei

- Status changed from WIP to Review

- % Done changed from 0 to 100

The changes are in 3821c rev 12835.

#6 - 08/18/2021 09:17 AM - Greg Shah

- Status changed from Review to Closed

Code Review Task Branch 3821c Revision 12835

The solution is elegant and it makes sense.

In public raw(byte[] val), I think this text needs to be removed from the javadoc: "The passed reference becomes the backing reference for this instance (the data is not copied, only the reference to the data is copied).".

#7 - 08/18/2021 10:13 AM - Constantin Asofiei

Greg Shah wrote:

> In public raw(byte[] val), I think this text needs to be removed from the javadoc: "The passed reference becomes the backing reference for this instance (the data is not copied, only the reference to the data is copied).".

Fixed in 3821c/12837.

#8 - 08/18/2021 12:57 PM - Constantin Asofiei

- Related to Bug #5608: raw vs memptr parameters added

Files

ca_upd20140813a.zip	65.7 KB	08/13/2014	Constantin Asofiei