Base Language - Feature #4064

large-raw data type

05/01/2019 04:29 PM - Greg Shah

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

Description

Related issues:

Related to Base Language - Feature #3254: add support for running 4GL on mult...	**WIP**

History

#1 - 05/01/2019 04:37 PM - Greg Shah

In the 4GL, the RAW type stores binary data up to 31000 bytes in size. For larger binary data, a MEMPTR must be used. In the 4GL, this is a real allocation of the host system's memory and can be as large as the free memory the 4GL process can allocate.

In the 4GL MEMPTR usage is in-process, but in FWD it is a client-side resource. When used for passing parameters to native API calls, such an implementation is needed. But if the application code is just using MEMPTR to store larger byte sizes than 31000, then the FWD approach is a costly way to implement.

LARGE-RAW will be a new kind of BDT (actually a subclass of BinaryData so it can be used anywhere a RAW or MEMPTR is used) which can store unlimited size data as a Java byte[]. This will be stored and manipulated in the FWD server, just like a RAW.

This task is intended to create this new BDT class AND to create the conversion support that allows new variables of this type to be created.

There is a potential performance improvement that comes with this feature. Any large binary data usage that is safely implemented only on the server side, can be switched to LARGE-RAW for a speedup.

#2 - 05/02/2019 09:43 AM - Constantin Asofiei

Do you expect for the 4GL programs to be re-written using the large-raw data-type? Because I would use hints at conversion time and automatically replace memptr with large-raw in a per-file/folder hint.

Also, if we want this to be a server-side replacement for memptr, I would make it to extend memptr, and not BinaryData - otherwise we will break any 4GL APIs expecting memptr and only memptr. At least, we need to check where we have memptr hard-coded as a parameter type for 4GL APIs.

#3 - 05/02/2019 10:15 AM - Greg Shah

> Do you expect for the 4GL programs to be re-written using the large-raw data-type? Because I would use hints at conversion time and

> automatically replace memptr with large-raw in a per-file/folder hint.

I want to make sure that at a minimum, future 4GL code can use this type directly.

I'm open to hints, but I suspect it would be possible to calculate if there is any unsafe usage of a given MEMPTR. Instances that have no unsafe usage could be automatically converted to LARGE-RAW.

The primary unsafe usage is passing to or returning from a native API call. We could consider that copying to or from another MEMPTR which itself is used unsafely, could also be a reason to leave it as a MEMPTR, but it is not inherently unsafe to do this.

> if we want this to be a server-side replacement for memptr, I would make it to extend memptr, and not BinaryData - otherwise we will break any 4GL APIs expecting memptr and only memptr.

Agreed, this is a good idea.

#4 - 05/02/2019 10:19 AM - Greg Shah

In subclassing from MEMPTR, we might want to use a name like SERVER-MEMPTR. Or we can add an ON-SERVER option to the DEF VAR and leave the 4GL type name alone.

#5 - 07/03/2019 04:41 PM - Greg Shah

- Related to Feature #3254: add support for running 4GL on multiple threads in a single session added