Base Language - Feature #1624

Feature # 1623 (Closed): refine I/O support to resolve incompatibilities or missing features

add support for PUT CONTROL

10/21/2012 09:53 AM - Greg Shah

Status:	Closed	**Start date:**	01/21/2014
Priority:	Normal	**Due date:**	01/24/2014
Assignee:	Evgeny Kiselev	**% Done:**	100%
Category:		**Estimated time:**	8.00 hours
Target version:	Cleanup and Stablization for Server Features		
billable:	No	**vendor_id:**	GCD
Description			

History

#1 - 10/31/2012 01:42 PM - Greg Shah

- Target version set to Milestone 7

#2 - 01/14/2013 06:36 PM - Greg Shah

This is already supported. Someone should generate some representative testcases and prove that there are no issues, then this task can be closed.

#3 - 04/25/2013 11:11 AM - Greg Shah

- Target version changed from Milestone 7 to Milestone 11

#4 - 05/02/2013 06:56 PM - Greg Shah

- Parent task set to #1623

- Assignee set to Evgeny Kiselev

- Due date set to 08/12/2013

- Start date set to 08/10/2013

#5 - 05/02/2013 07:04 PM - Greg Shah

- Due date changed from 08/12/2013 to 08/13/2013

#6 - 01/21/2014 11:24 AM - Greg Shah

- Due date changed from 08/13/2013 to 01/24/2014

- Start date changed from 08/10/2013 to 01/21/2014

#7 - 01/31/2014 06:45 AM - Evgeny Kiselev

- Status changed from New to WIP

#8 - 01/31/2014 08:03 AM - Evgeny Kiselev

Is this test correct ?

```
def var i1 as integer.
def var i2 as integer.
def var i3 as integer.
def var c1 as character.

i1 = 1231221.
i2 = 1231.
i3 = 512412251.
```

```
c1 = "bla ~033[asgdsg".

output to "test_put_control.txt".
put control i1 chr(29) i2 chr(29) i3 chr(29).
put control c1.
output close.

def var m  as memptr    no-undo.
def var c as character no-undo.

set-size( m ) = 100.
input from "test_put_control.txt" binary no-convert.
import unformatted m.
input close.

c = get-string( m, 1 ).
if entry( 1, c, chr(29)) <> string(i1) then message "error 1".
if entry( 2, c, chr(29)) <> string(i2) then message "error 2".
if entry( 3, c, chr(29)) <> string(i3) then message "error 3".
if entry( 4, c, chr(29)) <> c1 then message "error 4".
```

#9 - 01/31/2014 09:20 AM - Greg Shah

It is close. Some things to add:

1. I think the testing code is too dependent upon using a delimiter. In a scenario where we are writing binary values, we can use tools to read the binary values more directly.

2. Make sure you test all non-printable characters in the extended ASCII range (0 - 255).

I think a good first test would be to simply loop through all possible character values:

```
do i = 0 to 255:
   put control chr(i).
end.
```

Then read all these back in to a memptr (don't use IMPORT, but rather try using COPY-LOB which properly handles binary data). Then you can use GET-BYTE to read each byte value.

3. Make sure to add tests for the use of the NULL and the NULL syntax.

4. Make sure to add tests to use PUT CONTROL raw_var (instead of using character output).

5. I like your idea of output of multiple values in 1 PUT CONTROL statement. Just change the code that reads things back in to be less dependent upon delimiters. Use more direct access to the bytes to determine conformance.

#10 - 02/02/2014 07:15 PM - Evgeny Kiselev

updated tests.

```
def var i as integer.
def var i2 as integer.
def var lchar as longchar.
def var m  as memptr    no-undo.
def var c as character no-undo.

output to "test_put_control.txt".
do i = 0 to 255:
   put control chr(i).
end.
put control null.

output close.

set-size( m ) = 256.
copy-lob from file "test_put_control.txt" to m.

do i = 1 to 255:
   if get-byte(m, i) <> i then message "error " get-byte(m, i) " <> " i.
end.

if get-byte(m, 256) <> 0 then message "error NULL <> " get-byte(m, 256).
```

```
def var i1 as integer.
def var i2 as integer.
def var i3 as integer.
def var c1 as character.
def var c_res as character.
def var m  as memptr.
def var rvar as raw.

put-string(rvar, 1) = "test raw data".

i1 = 1231221.
i2 = 1231.
i3 = 512412251.
c1 = "bla ~033[asgdsg".

output to "test_put_control.txt".
put control i1 chr(29) i2 chr(29) i3 chr(29).
put control c1.
put control rvar.
output close.

output close.

set-size(m) = 256.
copy-lob from file "test_put_control.txt" to m.

c_res = string(i1) + chr(29) + string(i2) + chr(29) + string(i3) + chr(29).

if get-string(m, 1, 23) <> c_res then message "error '" get-string(m, 1, 23) "' <> '" c_res "'".
if get-string(m, 24, 12) <> c1 then message "error '" get-string(m, 24, 12) "' <> '" c1 "'".
if get-string(m, 36, 13) <> "test raw data"
          then message "error '" get-string(m, 36, 13) "' <> test raw data".
```

#11 - 02/03/2014 06:57 PM - Greg Shah

The first testcase above has a bug (you write out 257 bytes but size the memptr at 256).

Also, it doesn't test the NULL(x) syntax. Otherwise it seems useful.

Please document your findings about how this works in the 4GL and how P2J does for the same code.

#12 - 02/04/2014 07:58 PM - Evgeny Kiselev

- File results.zip added

```
def var i as integer.
def var i2 as integer.
def var lchar as longchar.
def var m  as memptr    no-undo.
def var c as character no-undo.

output to "test_put_control.txt".
do i = 1 to 255:
   put control chr(i).
end.
put control null.
put control c.

output close.

set-size( m ) = 257.
copy-lob from file "test_put_control.txt" to m.

do i = 1 to 255:
   if get-byte(m, i) <> i then message "error " get-byte(m, i) " <> " i.
end.

if get-byte(m, 256) <> 0 then message "error NULL <> " get-byte(m, 256).
if get-byte(m, 257) <> ? then message "error NULL <> " get-byte(m, 257).
```

First test working fine, except wrong conversion of copy-lob from file "test_put_control.txt" to m.. P2J convert it's in uncompilable code: LargeObjectOps.writeToFile("test_put_control.txt", m);
It should be read from file to memptr, and arguments for this method are swapped. So I've compare just files with result. In the first test result is identical:
test_put_control.txt   - generated by 4gl.
test_put_control_p2j.txt - generated by P2J.

Second test:

```
def var i1 as integer.
def var i2 as integer.
def var i3 as integer.
def var c1 as character.
def var c_res as character.
def var m  as memptr.
def var rvar as raw.

put-string(rvar, 1) = "test raw data".

i1 = 1231221.
i2 = 1231.
i3 = 512412251.
c1 = "bla ~033[asgdsg".

output to "test_put_control2.txt".
put control i1 chr(29) i2 chr(29) i3 chr(29).
put control c1.
put control rvar.
```

```
output close.

set-size(m) = 256.
copy-lob from file "test_put_control2.txt" to m.

c_res = string(i1) + chr(29) + string(i2) + chr(29) + string(i3) + chr(29).

if get-string(m, 1, 23) <> c_res then message "error '" get-string(m, 1, 23) "' <> '" c_res "'".
if get-string(m, 24, 12) <> c1 then message "error '" get-string(m, 24, 12) "' <> '" c1 "'".
if get-string(m, 36, 13) <> "test raw data"
          then message "error '" get-string(m, 36, 13) "' <> test raw data".
```

Gave incorrect results because of raw type.
02000EdGVzdCByYXcgZGF0YQA= P2J result vs test raw data in 4GL.(looks like BASE64 encoding)

Special cases:
put control null converted into \00 symbol
put control c where c is unknown then wrote nothing.

> Also, it doesn't test the NULL

Could you explain more about NULL(x) ? is this the same like put control null. ?

#13 - 02/05/2014 09:03 AM - Greg Shah

> except wrong conversion of copy-lob from file "test_put_control.txt" to m.. P2J convert it's in uncompilable code:

LargeObjectOps.writeToFile("test_put_control.txt", m);

COPY-LOB support is not complete. I had put support in for 1 form only (memptr to file) and it isn't proven to be compatible yet.

> It should be read from file to memptr, and arguments for this method are swapped.

No, the issue is that there needs to be a LargeObjectOps.readFromFile(String/character, BinaryData). Please add this and change the conversion to differentially emit the right method name based on which operation is needed. We don't need to do anything more extensive nor do we need to prove compatibility. I just want to make sure we have some basic support to allow testcases to work.

> Gave incorrect results because of raw type.
> 02000EdGVzdCByYXcgZGF0YQA= P2J result vs test raw data in 4GL.(looks like BASE64 encoding)

If I understand correctly, the problem is that our PUT CONTROL implementation converts the RAW to a STRING before output. This is incorrect. We need to just output the bytes directly. Please fix it (this is a problem with PUT CONTROL, not RAW).

> Special cases:
> put control null converted into \00 symbol
> put control c where c is unknown then wrote nothing.

Are you saying these are deviations in our implementation? If so, fix them.

> Could you explain more about NULL(x) ? is this the same like put control null. ?

The 4GL docs state that there is syntax like this:

```
PUT CONTROL NULL(5).
```

It states that this should result in 5 null bytes being written. Please test this with various values (including negative and 0).

#14 - 02/09/2014 08:09 PM - Evgeny Kiselev

- File put_control_testcases20140208a.zip added

- File evk_upd20140208a.zip added

1) Added conversion for COPY-LOB from file to memptr
2) Added implementation for COPY-LOB
3) Added for RAW type special implementation of PUT CONTROL

#15 - 02/11/2014 05:47 PM - Greg Shah

Code Review 0208a

1. Does the Steam.generateFormattedText() have implications when the statement is not PUT CONTROL? For example, how does the 4GL handle a regular PUT raw_var?

2. Should the Steam.generateFormattedText() change check for instanceof BinaryData instead of raw? In other words, can memptr also be used and does it have the same behavior?

3. Update copyright end dates to have 2014.

4. Check in your testcases.

#16 - 02/13/2014 07:45 PM - Evgeny Kiselev

Greg Shah wrote:

> Code Review 0208a
>
> 1. Does the Steam.generateFormattedText() have implications when the statement is not PUT CONTROL? For example, how does the 4GL handle a regular PUT raw_var?

put control rvar. works fine but put rvar. throw a compile time exception: INPUT/OUTPUT operations are not allowed with RAW, ROWID, MEMPTR, BLOB, CLOB or LONGCHAR type variables. (11382)

> 2. Should the Steam.generateFormattedText() change check for instanceof BinaryData instead of raw? In other words, can memptr also be used and does it have the same behavior?

No, The testcase does not compile:
put control memptr_var. and put control longchar_var.
throws a compile time exception: INPUT/OUTPUT operations are not allowed with RAW, ROWID, MEMPTR, BLOB, CLOB or LONGCHAR type variables. (11382)
But it work fine with RAW data type.

#17 - 02/14/2014 05:26 PM - Greg Shah

Update the copyright dates and post the final update here. Then start regression testing (both conversion and runtime).

Please also check in your testcases.

#18 - 02/14/2014 07:41 PM - Evgeny Kiselev

- File 01_evk_upd20140214a.zip added

Final update. Regression is running now.

#19 - 02/16/2014 03:20 AM - Evgeny Kiselev

Evgeny Kiselev wrote:

> Final update. Regression is running now.

Passed regression testing

#20 - 02/16/2014 08:01 AM - Greg Shah

Check it in and distribute it.

#21 - 02/18/2014 11:03 PM - Evgeny Kiselev

- File evk_upd20140214a.zip added

#22 - 02/18/2014 11:13 PM - Evgeny Kiselev

- Status changed from WIP to Review

- % Done changed from 0 to 100

- File evk_upd20140214a.zip added

#23 - 02/19/2014 11:52 AM - Evgeny Kiselev

update evk_upd20140214a.zip has been passed regression testing.

Committed to bzr revision 10470.

#24 - 02/19/2014 12:15 PM - Greg Shah

- Status changed from Review to Closed

#25 - 02/19/2014 11:00 PM - Evgeny Kiselev

Do I need to commit testcases ?

#26 - 02/20/2014 01:26 AM - Constantin Asofiei

Evgeny Kiselev wrote:

> Do I need to commit testcases ?

Of course, the testcases need to be in the repository.

#27 - 11/16/2016 12:07 PM - Greg Shah

- Target version changed from Milestone 11 to Cleanup and Stablization for Server Features

Files

results.zip	1.3 KB	02/05/2014	Evgeny Kiselev
evk_upd20140208a.zip	40.3 KB	02/10/2014	Evgeny Kiselev
put_control_testcases20140208a.zip	1016 Bytes	02/10/2014	Evgeny Kiselev
01_evk_upd20140214a.zip	40.3 KB	02/15/2014	Evgeny Kiselev
evk_upd20140214a.zip	40.3 KB	02/19/2014	Evgeny Kiselev
evk_upd20140214a.zip	40.3 KB	02/19/2014	Evgeny Kiselev