

Bugs - Bug #1702

message_stmt[3-5].p issues

10/29/2012 08:33 AM - Constantin Asofiei

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

History

#1 - 10/29/2012 08:34 AM - Constantin Asofiei

Following are some findings about the message stmt issues. Attached are the original description of this issue, some new tests and merged files with WIP changes.

Part 1:

The current message_stmt*.p testcases present several problems; so far, I've investigated the following issues:

1. message_stmt3.p - related to message ... update statement, when output is redirected to an unnamed stream. This issue is related to the ThinClient.message(String, BDT, boolean, String, int) method. If F4 key is pressed during update, there will be no output to the unnamed stream, regarding this message statement. So, the message must be output only to the screen and, in the finally block, if the update was OK, output the message text and the variable, if any, to the stream. Also, the variable must be formatted accordingly to its specified format, if any - this can be solved by saving the rendered text from the FillIn widget, as computed in TC.tinyInput.

2. message_stmt5.p - related to update statement with an editing block, when output is redirected to an unnamed stream. Currently, the update frame is not displayed at all. This I think is due to a "switchStreams(frame)" call in TC.promptFor - this will output the frame to the stream, instead of terminal.

The idea is that, if F4 is pressed during edit, there will be no output to the stream. The frame will output to the stream after the editing completes successfully.

So, a solution would be:

- if within in editing mode and input is not redirected, TC.promptFor will always output to the terminal.
- all the events generated within the editing block will be executed on the terminal. If the editing block finishes successfully, the frame will be output to the unnamed stream.

3. I could not find any situation where, when redirected input source is NOT /dev/null, there will be output to the terminal, in case when output is redirected via an unnamed stream. Even if INPUT FROM or INPUT THROUGH is used, there is no output to the terminal in this case.

Part 2:

Investigating the case when input is redirected to /dev/null, showed that this does not affect message statements. What it affects are the cases when input is needed to be read and /dev/null presents no such input. In this case, the program will fail silently:

input from /dev/null.
output to a.txt.

message "bar: " update x as char.

output close.
input close.

display "foo" with frame f.

This display will never be executed. If for the message statement above the UPDATE clause is removed, there will be output to the stream, but not to the terminal. If input is redirected to a file, output will be the same: not to the terminal, yes to the stream.

The other issues, about mixed input/output redirections require some changes to how TC.message(String, BDT, ...) method works. This method would require two additional parameters - one about the unnamed input redirection status and one about the NO-ECHO flag.

Based on these two flags, decisions will be made to whether output the message (and var) to the stream, terminal or both. There is one special case - if the message should output to the stream and not to the terminal, the variable still needs to be formatted accordingly. For this, TC.tinyInput must be

modified to be able to return a MessageReturnValue (which will contain the formatted BDT), but without the wait-for.

Finally, I still have some concerns about the "switchStreams(frame)" call in TC.promptFor. Currently, the client has no knowledge about the unnamed input redirected status (all this is handled on the server-side). So, I think it is needed to investigate how UPDATE/SET/PROMPT-FOR when input and output is redirected to an unnamed stream. My first guess is that they might behave similar to the MESSAGE ... UPDATE statement.

#2 - 10/29/2012 08:35 AM - Constantin Asofiei

On 10/25/2012 02:20 PM, Vadim wrote:

Hello,

During investigation of bug [#18](#) I have noticed the difference in behavior Progress 4GL and converted code.

If I add

OUTPUT TO VALUE.

Progress 4GL redirects output fully to file. Converted code creates specified file, but actually doesn't redirect messages: they are shown on terminal, and file became empty.

I tried to debug but not found the reason so far. Terminal to file stream redirection seems works properly. Should I investigate it too?

(I commented "native" in ant target "all" temporary, will investigate problem later)

Regards,

Vadim

#3 - 10/29/2012 08:36 AM - Constantin Asofiei

On 26.10.2012 00:54, Greg Shah wrote:

Vadim,

This behavior was found to be correct (or mostly correct) in character mode on Unix. Constantin had found a range of problems with our message statement, but the idea that the message statement was not always redirected to a file is something that is known to be correct in many cases. We can't make that claim for the 4GL under Windows (either GUI or CHUI).

A question: Did you test on both GUI and CHUI? If not, please do.

Constantin: can you point Vadim at the code that implements this? I know I have some emails from you that describe our remaining problems. We really should create Redmine issues for those. But it also seems Vadim may have found differences with the behavior on Windows.

Thanks,
Greg

#4 - 10/29/2012 08:36 AM - Constantin Asofiei

Hello,

I have tested using OUTPUT TO VALUE for Progress 4GL (CHIU and GUI) and for P2J ChUI Client.
Please check results below. Progress 4GL has different behavior for CHUI and GUI. P2J ChUI Client behavior exactly repeats CHUI behavior.

Progress 4GL

Character client Output both to terminal and file. File is closed after application finished (until that file exists with zero size)
Client (Procedure editor) Output only to file. Frame is not shown at all (frame with terminal is shown if OUTPUT TO VALUE... is not set)
P2J ChUI Client Output both to terminal and file. File is closed after application finished (until that file exists with zero size)

I also have found key place: ThinClient.sendToStream(String message) with the comment:

```
// the message is displayed on the terminal only if the input is not  
// read from a null device, like /dev/null
```

Regards,
Vadim

#5 - 10/29/2012 08:37 AM - Constantin Asofiei

Vadim,

Please look at the LogicalTerminal.messageMessage(Object[] list, boolean set, Accessor var, boolean auto, String format, ColorSpec cs) method too.

#6 - 10/29/2012 08:39 AM - Constantin Asofiei

- File *ca_test20091207a.zip* added

Added file with testcases and P2J code from back in 2009.

#7 - 10/29/2012 09:17 AM - Greg Shah

- Assignee set to *Vadim Nebogatov*

#8 - 10/31/2012 02:24 PM - Greg Shah

- Target version set to *Milestone 7*

#9 - 04/25/2013 11:01 AM - Greg Shah

- Target version changed from *Milestone 7* to *Milestone 11*

#10 - 02/28/2014 04:55 PM - Greg Shah

A problem found in #2228 (note 12) has highlighted problems with SKIP, embedded newlines and also redirected mode. A quick fix has been created that fixes the SKIP and embedded newlines almost 100% for the terminal. What remains is the redirected mode implementation (of multi-line output)

and a difference in showing an extra space in reverse video on some terminal output.

Please see `uast/message_stmt_skip.p` to recreate the problems (there are 4 modes it can be run in).

#11 - 02/28/2014 05:02 PM - Greg Shah

See note 163 in #2228 for testcase and the update that resolves most of the SKIP issues. Please note that the testcase has also been checked into bzt.

#12 - 05/12/2014 04:02 PM - Vadim Nebogatov

- Status changed from New to WIP

#13 - 06/06/2014 05:34 PM - Vadim Nebogatov

Result of conversion `message_stmt5.p` is not compilable:

```
[javac] /home/vmn/p2j/testcases/uast/src/com/goldencode/testcases/ui/TestFrame0.java:11: error: getX() in
TestFrame0 clashes with getX() in CommonFrame
[javac]     public character getX();
[javac]           ^
[javac]     return type character is not compatible with NumberType
```

#14 - 06/10/2014 03:31 AM - Constantin Asofiei

Vadim Nebogatov wrote:

```
Result of conversion message_stmt5.p is not compilable:
getX() in TestFrame0 clashes with getX() in CommonFrame
```

This is a collision in the generated APIs and the CommonFrame APIs - to get past this, rename the variable named `x` to something else, in the testcase; the problem you are seeing is part of a larger problem, which is seen in buffer field names, too (converted names collide with APIs defined in interfaces).

#15 - 07/11/2014 10:02 AM - Greg Shah

- Target version changed from Milestone 11 to Milestone 12

#16 - 03/23/2016 02:20 PM - Greg Shah

- Assignee deleted (Vadim Nebogatov)

- Target version deleted (Milestone 12)

Files

<code>ca_test20091207a.zip</code>	280 KB	10/29/2012	Constantin Asofiei
-----------------------------------	--------	------------	--------------------