

Base Language - Feature #1628

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

fix any incompatibilities or missing features of NO-ECHO support

10/21/2012 10:15 AM - Greg Shah

Status:	Closed	Start date:	01/27/2014
Priority:	Normal	Due date:	01/31/2014
Assignee:	Constantin Asofiei	% Done:	100%
Category:		Estimated time:	24.00 hours
Target version:	Cleanup and Stablization for Server Features	version:	
billable:	No		
vendor_id:	GCD		
Description			
Related issues:			
Related to User Interface - Bug #2345: already visible frames will sometimes ...			New

History

#1 - 10/31/2012 01:41 PM - Greg Shah

- Target version set to Milestone 7

#2 - 04/25/2013 11:14 AM - Greg Shah

- Target version changed from Milestone 7 to Milestone 11

#3 - 05/02/2013 07:02 PM - Greg Shah

- Assignee set to Evgeny Kiselev
- Start date set to 08/14/2013
- Estimated time changed from 12.00 to 24.00
- Due date set to 08/19/2013
- Parent task set to #1623

#4 - 05/02/2013 07:05 PM - Greg Shah

- Due date changed from 08/19/2013 to 08/26/2013
- Start date changed from 08/14/2013 to 08/17/2013

#5 - 01/21/2014 11:29 AM - Greg Shah

- Due date changed from 08/26/2013 to 01/31/2014
- Start date changed from 08/17/2013 to 01/27/2014

Based on the NO-ECHO testcases created in #1629, test the P2J implementation for NO-ECHO and make any functional additions or fixes as needed.

#6 - 03/07/2014 04:23 PM - Greg Shah

Make sure you have a testcase that does something like this:

```
INPUT THROUGH echo $USER NO-ECHO.
SET the-current-user AS CHARACTER.
INPUT CLOSE.
MESSAGE the-current-user.
```

#7 - 03/10/2014 06:51 PM - Evgeny Kiselev

Is any way to check what display is showing now in 4gl ?

#8 - 03/10/2014 07:08 PM - Greg Shah

What do you mean by "display"?

#9 - 03/10/2014 07:10 PM - Evgeny Kiselev

For example if I set ECHO I want to know in code if something is displayed and if I set NO-ECHO then display buffer is empty.

#10 - 03/10/2014 08:04 PM - Greg Shah

OK, I understand what you are trying to do.

I think there are 2 options (test these to see if either or both work):

1. Try redirecting the terminal output to a file:

OUTPUT TO echo_test.log.

2. Try redirecting STDOUT for the 4GL process:

pro 1> echo_test_stdout.log

This is less desirable than the OUTPUT TO because to use the same technique in P2J the user running the testcase must know/remember to execute the client with STDOUT redirected.

From there, you can examine the log file and look for specific entries. Make each line of your INPUT source unique so that you can tell which statements generate output or not.

If you use different filenames for the log statements for each INPUT case, then you could use the 4GL to detect if those files existed and if they have a non-zero size.

Constantin: any other ideas?

#11 - 03/10/2014 08:46 PM - Greg Shah

I'm responding to the note in #1629.

Am correct ?

Yes. Although, I am surprised it doesn't work (we support other define var cases that are in a format phrase).

Also if I run testcases not from terminal (for example for IDE or other sources) than I got: ** Process launch failed.. (-2), but I think it's normal situation.

I suspect you are right.

#12 - 03/11/2014 07:43 PM - Evgeny Kiselev

Greg Shah wrote:

Yes. Although, I am surprised it doesn't work (we support other define var cases that are in a format phrase).

Do I need to look it now ? or it's not part of this task ?

#13 - 03/12/2014 08:46 AM - Greg Shah

No. But please do create a new Redmine task (a bug) that describes the problem and has the testcase code for the recreate. Make me a "watcher" on that task. Create the task in this "Base Language" project.

#14 - 04/23/2014 07:14 PM - Evgeny Kiselev

After investigation about problem in display while stream is opened, I've found next problem:

- 1) if output stream is opened, then in 4gl (and p2j) every display goes to this output stream.
- 2) after the stream is closed and if no any other display call into this frame, then all displayed data (after stream is opened, but before stream is closed) will not be shown. In P2J this work properly.
- 3) but if try to display after stream is closed, then all data will shown. I think it's redraw (refresh) all data into frame.

#15 - 04/24/2014 08:36 AM - Greg Shah

- 3) but if try to display after stream is closed, then all data will shown. I think it's redraw (refresh) all data into frame.

Please post the example code and output here.

#16 - 04/24/2014 11:11 AM - Evgeny Kiselev

Greg Shah wrote:

- 3) but if try to display after stream is closed, then all data will shown. I think it's redraw (refresh) all data into frame.

Please post the example code and output here.

testcase:

```
def var flag as logical.  
def var displayFrame as logical.  
  
message "display frame" update displayFrame.  
  
display "before 'input from'".  
  
output to value ("test.txt") append.  
  
put unformatted "~n ".  
  
display "something between".  
  
put unformatted "this is the appended text".  
  
if displayFrame then  
    display "some test,".  
  
output close.  
message "output close" view-as alert-box.  
  
if displayFrame then  
    display "after close".
```

Results in 4gl:
displayFrame flag = yes

```
|before 'input from' something between some test, after close|
```

displayFrame flag = no

```
|before 'input from'|
```

P2J results:
displayFrame flag = yes

```
|before 'input from'                                after close|
```

displayFrame flag = no

```
|before 'input from'|
```

#17 - 04/24/2014 01:32 PM - Greg Shah

Interesting.

It seems to me that all 4 DISPLAY statements are being output in the same "default" (unnamed) frame.

The first DISPLAY is before redirection occurs. It draws the frame on the screen and draws the first literal text widget. There is space left behind for the other possible literal strings which the layout manager would plan for in the order of definition.

I think that the redirection after the first display causes the strange behavior here. Even though the display "something between". code is unconditional (no if displayFrame then) the output is placed in the frame but it is somehow buffered or the frame is not being refreshed.

That buffered or unviewed content is then "flushed" to the already viewed screen only if there is another DISPLAY (or probably any VIEW) of that frame AFTER redirection is turned off.

If the frame wasn't already displayed when the redirection was started, then this behavior would not occur.

Note the difference in output to the test.txt file between these cases:

displayFrame eq yes case:

```
before 'input from' something between
this is the appended text
```

```
before 'input from' something between some test,
```

displayFrame eq no case:

```
before 'input from' something between
this is the appended text
```

This strange output behavior is due to the flushing that occurs when you mix frame output and stream output to the same destination. Does P2J handle the non-terminal portions OK?

The thing that we don't do today in the terminal portion is that the output done to that frame during the redirected period is not made visible in the interactive frame. I suspect this may have something to do with how we "switch" the screen rendering (see OutputPrimitives and RedirectedTerminal classes) for a special redirected mode renderer. We will have to make special provisions to honor this behavior.

Go ahead and fix this.

- File evk_upd20140502a.zip added

This is pre review update, without doc headers.

- 1) DirStream.java there is unnecessary new line. I'm not sure if this line needed at all, maybe it was needed only for windows version. So I've just move it into if block. Or maybe we don't need this line at all.
- 2) StreamFactory.java fix problem in #1629 in note 17 at point 2).
- 3) GenericFrame.java fix problem #1629 in note 17 at point 10),11)
- 4) ThinClient.java fix problem #1629 in note 17 at point 11.a). If message goes after message box, then in file this message shifted to the right(I think it shifted to the last position of cursor).

testcase for 1) and 2)

```
def var txt as character init "NOT test".
def var flag as logical.

message "Use NON-ECHO?" update flag.

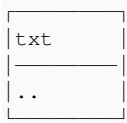
if flag then
    input from os-dir("") no-echo.
else
    input from os-dir("").

set txt.

if txt:screen-value <> "" then message "txt is properly set to " txt.
else message "something went wrong".

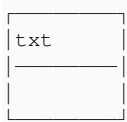
input close.
```

4gl result:



A screenshot of a 4GL window. The window has a title bar and a content area. The content area shows the variable 'txt' followed by a horizontal line and then the value '..'.

Before fix:



A screenshot of a 4GL window. The window has a title bar and a content area. The content area shows the variable 'txt' followed by a horizontal line and then the value '..'.

After fix:



A screenshot of a 4GL window. The window has a title bar and a content area. The content area shows the variable 'txt' followed by a horizontal line and then the value '..'.

testcase for 3) and 4)

```
def var flag as logical.
def var displayFrame as logical.
```

```
flag = yes.  
displayFrame = yes.  
  
message "display frame" update displayFrame.  
  
display "before 'input from'".  
  
output to value ("test.txt") append echo.  
  
display "some between".  
message "test message".  
output close.  
  
message "output close" view-as alert-box.  
  
if displayFrame then display "after close".  
  
message "end" view-as alert-box.
```

#19 - 05/05/2014 11:47 AM - Greg Shah

Code Review 0502a

1. Is the GenericFrame change intentionally meant to only modify the case where DISPLAY is used with unnamed stream redirection (and not when DISPLAY is used with named streams)?
2. Please provide some more description of your intentions with the changes in GenericFrame.view(FrameElement[]). It looks like you are forcing all widgets in the frame to be displayed in this case. I wonder if there are scenarios where there are widgets in this list which should not be displayed.
3. Does GenericFrame.idsFromWidgets() do the same thing as you are doing in view()? If so, you can reuse that code.
4. In regard to the NEWLINE in DirStream, please test on windev01 to see if it is needed.

#20 - 05/05/2014 09:49 PM - Evgeny Kiselev

Greg Shah wrote:

Code Review 0502a

1. Is the GenericFrame change intentionally meant to only modify the case where DISPLAY is used with unnamed stream redirection (and not when DISPLAY is used with named streams)?

will check it.

2. Please provide some more description of your intentions with the changes in `GenericFrame.view(FrameElement[])`. It looks like you are forcing all widgets in the frame to be displayed in this case. I wonder if there are scenarios where there are widgets in this list which should not be displayed.

Yes, I'm forcing all widget in frame to "refresh" their state. Best solution is remember only widget which have been displayed after stream open, but it'll be more complicated. I'm trying to find some tests if some widget should be not displayed.

3. Does `GenericFrame.idsFromWidgets()` do the same thing as you are doing in `view()`? If so, you can reuse that code.

Yes, I'll fix it.

4. In regard to the `NEWLINE` in `DirStream`, please test on `windev01` to see if it is needed.

`NEWLINE` is not needed for windows server

#21 - 05/09/2014 03:58 PM - Evgeny Kiselev

- File `evk_upd20140509a.zip` added

Behaviour of display and stream functionality is more complicated than I thought before.
Some display call can refresh widgets (either if display in different stream). display stream (named stream) in some situation will refresh widgets in other not. For example:

```
def var flag as logical.  
def var displayFrame as logical.  
define stream scr.  
output stream scr to terminal.  
define stream rep.  
output stream rep to FieldSizeReport.txt.  
flag = yes.  
displayFrame = yes.  
message "display frame" update displayFrame.  
display "1)first msg".  
output to value ("test.txt") append echo.
```



```
display stream scr "2) second msg".
display "3) third msg".
message "test message".
output close.
message "output close" view-as alert-box.
if displayFrame then display stream rep.
message "end" view-as alert-box.
```

4gl result:

```
|1)first msg 2) second msg|
```

But if we change rep to scr stream:

```
|1)first msg 2) second msg 3) third msg|
```

I couldn't find any way to find difference between src stream and rep stream. Maybe output stream scr to terminal. is not properly converted. `scrStream.isTerm()` return false for scr stream, but I'm not sure should it return true in this situation ?

#22 - 05/12/2014 02:58 PM - Greg Shah

Behaviour of display and stream functionality is more complicated than I thought before.

Yes, it is very tricky. It took us a long time to get it "mostly right". I know it is not completely correct right now, but it is more correct than incorrect.

In regard to your testcase and results, I believe that it is reasonably consistent. I will annotate the code to explain what I think is going on:

```
def var flag as logical.
def var displayFrame as logical.

/* at this point, there are no named streams and the unnamed stream is equivalent to the terminal */

define stream scr.
output stream scr to terminal.
```

```

/* now there is a named stream that also is equivalent to the terminal */
/* output to either the unnamed stream or to scr should be the same */

define stream rep.
output stream rep to FieldSizeReport.txt.

/* now there is another named stream, but it represents a file */

flag = yes.
displayFrame = yes.
message "display frame" update displayFrame.

display "1)first msg". /* A: this line outputs the default (unnamed) frame to the unnamed stream which is the
    terminal, at this point only the 1st static text widget is visible in the "screen-buffer" */

output to value ("test.txt") append echo.

/* now the unnamed stream is redirected to a file */

display stream scr "2) second msg". /* B: this line outputs the default (unnamed) frame to the named stream wh
    ich is the terminal, at this point both the 1st and 2nd static text widgets are visible in the "screen-buffer"
    */

display "3) third msg". /* C: this line outputs the default (unnamed) frame to the unnamed stream which is the
    file test.txt, at this point all 3 static text widgets are visible in the "screen-buffer", I guess the output
    file should have all 3 in it? */

message "test message".
output close.

/* the unnamed stream is now the terminal again */

message "output close" view-as alert-box.

if displayFrame then display stream rep. /* D: this line outputs the default (unnamed) frame to the file Field
    SizeReport.txt, no new output to the terminal here but all 3 static text widgets would get output to the file
    */

/*
if displayFrame then display stream scr. E: this line would output the default (unnamed) frame to the terminal
, all 3 static text widgets in the screen-buffer would get output to the terminal
*/

message "end" view-as alert-box.

```

Are all my notes correct based on your file outputs?

It is important to know that all your display statements are creating a single unnamed (default) frame. This default frame has a single screen-buffer, whose current state is output to whatever destination is specified in the output statement (e.g. DISPLAY in this case). In other words, the screen-buffer exists at a frame level and does not have different versions for each destination.

scrStream.isTerm() return false for scr stream, but I'm not sure should it return true in this situation ?

I would have expected it to return true once the output stream scr to terminal executes. There may be something wrong with the code in StreamFactory.openFileStream() or in our conversion.

#23 - 05/12/2014 06:21 PM - Evgeny Kiselev

Everything is correct, except one:

C: this line outputs the default (unnamed) frame to the unnamed stream which is the file test.txt, at this point all 3 static text widgets are visible in the "screen-buffer", I guess the output file should have all 3 in it?

In the end, all 3 widgets exists in file, but not after this line. (I think after stream is closed). As I know display is not supported unbuffered statement. So I think after close all data is flushed.

File output for this testcase:

```
1)first msg 2) second msg 3) third msg
test message
```

#24 - 05/13/2014 08:03 AM - Greg Shah

The client code will have to be looked at closely in regard to flushing. There are other times when flushing occurs besides closing the stream. For example, when you switch the stream to which you output a frame, it triggers a flush of the stream from which you are switching away.

#25 - 06/09/2014 08:58 AM - Evgeny Kiselev

- File evk_upd20140607a.zip added

Fixed all issues from note [#22](#), [#23](#), [#24](#)

testcases:

```
def var flag as logical.
def var displayFrame as logical.

message "Use NON-ECHO?" update flag.
message "display frame" update displayFrame.

if displayFrame then
  display "before 'input from'".

if flag then
  output to value ("test.txt") append echo.
else
  output to value ("test.txt") append.

put unformatted "~n ".

display "some test,".
message "message:test".
put unformatted "this is the appended text".

if displayFrame then
  display "some test,".

output close.
```

```

message "output close" view-as alert-box.

if displayFrame then
    display "after close".

def var flag as logical.
def var displayFrame as logical.
define stream scr.
output stream scr to terminal.
define stream rep.
output stream rep to FieldSizeReport.txt.
flag = yes.
displayFrame = yes.
message "display frame" update displayFrame.
message "display flag" update flag.

display "1)first msg".

output to value ("test.txt") append echo.

display stream scr "2) second msg".

display "3) third msg".

message "test message".
output close.
message "output close" view-as alert-box.
if displayFrame then display stream rep.
if flag then display "test".
message "end" view-as alert-box.

```

#26 - 06/09/2014 11:33 AM - Greg Shah

Code Review 0607a

1. GenericFrame is missing a history entry.
2. Can GenericFrame.refreshWidgets() could use Utils.integerCollectionToPrimitive() instead of the hand-coded approach?
3. The javadoc for GenericFrame.refreshWidgets() seems to be incorrect.
4. Can GenericFrame.cacheWidgetIDs() use Collections.addAll() instead of the hand-coded approach?

5. The `GenericFrame.view(FrameElement[] data)` use of `cacheWidgetIDs()` seems like it will never do anything. That method has a short-cut at the top that is executed when `UnnamedStreams.output() != null`, which means that in that case the `cacheWidgetIDs()` will not be executed. The `cacheWidgetIDs()` will only trigger when `UnnamedStreams.output() == null` but in that case, `cacheWidgetIDs()` will return without doing anything.

6. There are cases where `GenericFrame.refreshWidgets()` is called many times during one `display()` invocation. For example, `GenericFrame.display()` will trigger `refreshWidgets()` at least 3 times because `refreshWidgets()` is invoked directly in that method as well as in both downstream methods that are called: `GenericFrame.display(FrameElement[])` and `GenericFrame.display(Stream, FrameElement[])` (when unnamed output is redirected). This is confusing at a minimum. The code is protected from actually calling down to the client, but it seems like there are cases where the call is unnecessary (`GenericFrame.display()` seems not needed ever).

7. It is not clear that the approach is correct. Some questions:

- Why is the caching being done before some calls to `LogicalTerminal.view()` but not for others?
- The caching of IDs occurs whenever `UnnamedStreams.output() != null`, but the output that is being done may be targeted at either `UnnamedStreams.output()` OR some other completely different output destination. It is not clear if all of these cases should trigger the caching. My suspicion is that it should be more limited.
- The `refreshWidgets()` is done during display statements, but it is not clear why/when it is needed. Should this only be done when the output destination is changing or is there some other limiting factor that comes into play?
- Is the `refreshWidgets()` only supposed to be done for `DISPLAY` and not for `VIEW` statements?

#27 - 06/09/2014 08:26 PM - Evgeny Kiselev

Greg Shah wrote:

Code Review 0607a

1. `GenericFrame` is missing a history entry.
2. Can `GenericFrame.refreshWidgets()` could use `Utils.integerCollectionToPrimitive()` instead of the hand-coded approach?
3. The javadoc for `GenericFrame.refreshWidgets()` seems to be incorrect.
4. Can `GenericFrame.cacheWidgetIDs()` use `Collections.addAll()` instead of the hand-coded approach?

fixed, will provide in next update

5. The `GenericFrame.view(FrameElement[] data)` use of `cacheWidgetIDs()` seems like it will never do anything. That method has a short-cut at the top that is executed when `UnnamedStreams.output() != null`, which means that in that case the `cacheWidgetIDs()` will not be executed. The `cacheWidgetIDs()` will only trigger when `UnnamedStreams.output() == null` but in that case, `cacheWidgetIDs()` will return without doing anything.

You are right, I've been added code here just for "safe", if this code will changed later.

6. There are cases where `GenericFrame.refreshWidgets()` is called many times during one `display()` invocation. For example, `GenericFrame.display()` will trigger `refreshWidgets()` at least 3 times because `refreshWidgets()` is invoked directly in that method as well as in both downstream methods that are called: `GenericFrame.display(FrameElement[])` and `GenericFrame.display(Stream, FrameElement[])` (when unnamed output is redirected). This is confusing at a minimum. The code is protected from actually calling down to the client, but it seems like there are cases where the call is unnecessary (`GenericFrame.display()` seems not needed ever).

Yes, it was not always clear where we need to save IDs, I will recheck it again.

7. It is not clear that the approach is correct. Some questions:

- Why is the caching being done before some calls to `LogicalTerminal.view()` but not for others?

I did not cache only in two places with `frameID` as widget `LogicalTerminal.view(frameID)`. As I understand all frame cannot be hidden by different stream.

- The caching of IDs occurs whenever `UnnamedStreams.output()` != null, but the output that is being done may be targeted at either `UnnamedStreams.output()` OR some other completely different output destination. It is not clear if all of these cases should trigger the caching. My suspicion is that it should be more limited.

Don't know at the moment, need research this.

- The `refreshWidgets()` is done during display statements, but it is not clear why/when it is needed. Should this only be done when the output destination is changing or is there some other limiting factor that comes into play?

Widgets will shown only after stream is close and if any new display command will run. If output destination when nothing happened. see testcases from note-25

- Is the `refreshWidgets()` only supposed to be done for `DISPLAY` and not for `VIEW` statements?

`refreshWidgets()` supposed to be done only for `DISPLAY`. `VIEW` statement doesn't do anything in this situation.

#28 - 06/10/2014 11:02 AM - Greg Shah

5. The `GenericFrame.view(FrameElement[] data)` use of `cacheWidgetIDs()` seems like it will never do anything. That method has a short-cut at the top that is executed when `UnnamedStreams.output()` != null, which means that in that case the `cacheWidgetIDs()` will not be executed. The `cacheWidgetIDs()` will only trigger when `UnnamedStreams.output()` == null but in that case, `cacheWidgetIDs()` will return without doing anything.

You are right, I've been added code here just for "safe", if this code will be changed later.

Please put a `//` comment there that explains this (that it should not be executed but is being placed there for safety).

- Why is the caching being done before some calls to `LogicalTerminal.view()` but not for others?

I did not cache only in two places with `frameID` as widget `LogicalTerminal.view(frameID)`. As I understand all frame cannot be hidden by different stream.

What do you mean by all frame cannot be hidden by different stream?

At a minimum, please put `//` comments in those locations explaining that the `cacheWidgetIDs()` is not needed there. That way future readers won't think it was mistakenly left out.

- The `refreshWidgets()` is done during display statements, but it is not clear why/when it is needed. Should this only be done when the output destination is changing or is there some other limiting factor that comes into play?

Widgets will be shown only after stream is closed and if any new display command will run. If output destination when nothing happened. see testcases from note-25

I don't see how these limitations are coded. There are no checks for these conditions.

- Is the `refreshWidgets()` only supposed to be done for `DISPLAY` and not for `VIEW` statements?

`refreshWidgets()` supposed to be done only for `DISPLAY`. `VIEW` statement doesn't do anything in this situation.

Please add text to the `refreshWidgets()` javadoc explaining that the 4GL only triggers this behavior from `DISPLAY` and does not trigger it from `VIEW`.

#29 - 06/11/2014 07:19 PM - Evgeny Kiselev

I need to determine Terminal stream. But got a problem:

```
define stream scr.  
output stream scr to terminal.
```

This 4gl code converted to:

```
TransactionManager.registerTopLevelFinalizable(scrStream, true);  
scrStream.assign(null);
```

scrStream is StreamWrapper and isTerm() method always return false.

I need to know is Stream is terminal or not, because display stream scr "some" works different if stream is not a terminal.

In code I'm trying to determine terminal like:

```
boolean isTerm = stream == null ||  
                stream instanceof StreamWrapper &&  
                ((StreamWrapper) stream).dereference() == null ||  
                stream.isTerm();
```

Is it correct ?

#30 - 06/12/2014 09:02 AM - Greg Shah

- File ges_upd20140612a.zip added

I think our approach to explicit terminal stream processing has been poor in the past. We "took advantage" of using null but this resulted in very confusing code.

I believe that the attached update is a better approach. I hope it is safe, easier to understand and more correct for your case. It compiles but I have NOT tested it.

Please check if this is working. Does this resolve your problem? Does this work for the general cases where there is an explicit redirection of the terminal:

```
OUTPUT TO TERM.  
OUTPUT TO TERMINAL.  
OUTPUT TO "tErM".  
OUTPUT TO "termiNAL".  
OUTPUT STREAM my-stream TO TERM.  
OUTPUT STREAM my-stream TO TERMINAL.  
OUTPUT STREAM my-stream TO "tErM".
```


OUTPUT STREAM my-stream TO "termiNAL".
INPUT FROM TERM.
INPUT FROM TERMINAL.
INPUT FROM "tErM".
INPUT FROM "termiNAL".
INPUT STREAM my-stream FROM TERM.
INPUT STREAM my-stream FROM TERMINAL.
INPUT STREAM my-stream FROM "tErM".
INPUT STREAM my-stream FROM "termiNAL".

If this works properly, then please include this with your code. It will go into testing at the same time.

Constantin: please do a code review of this. I don't see any cases of this idiom (`myStreamWrapper.assign(null)`) being used in the TIMCO srcnew/ code, so I think it should be safe.

#31 - 06/15/2014 08:23 PM - Evgeny Kiselev

Greg Shah wrote:

Please check if this is working. Does this resolve your problem? Does this work for the general cases where there is an explicit redirection of the terminal:

Yes it resolved my problem, and now everything are working fine. I've just finishing testing and upload update soon.

#32 - 06/16/2014 08:07 PM - Evgeny Kiselev

- File *evk_upd20140616a.zip* added

This update merged with *ges_upd20140612a.zip*

Fixed issues from note 28.

Also I've decided to remove unnecessary calls to `refreshWidgets()` method. Now it calls only once.

All my tests worked fine with update *ges_upd20140612a*.

#33 - 06/17/2014 07:15 AM - Greg Shah

Code Review 0616a

I'm OK with the changes. Please run both conversion and runtime regression testing. If you haven't run conversion on `devsrv01` recently, you will have to run without your changes to get a fresh baseline for comparison purposes.

This update does change code in a very sensitive area (the `GenericFrame` changes). I am a bit worried about that part. As such, screen and report failures will have to be examined very carefully.

#34 - 06/17/2014 07:16 AM - Greg Shah

Constantin: do you have any concerns with my changes (or with Evgeny's)?

#35 - 06/17/2014 08:48 AM - Constantin Asofiei

Greg Shah wrote:

Constantin: do you have any concerns with my changes (or with Evgeny's)?

The changes look OK to me.

#36 - 06/17/2014 09:14 PM - Evgeny Kiselev

Regression testing is finished. But atm I've found quite a lot timeout failed tests:

50 in gso_tests

42 in tc_tests

I've run tests twice time but most of them was failed by timeout in both tries. Now I'm running regression without update to see difference, maybe I was just unlucky.

#37 - 06/24/2014 07:54 PM - Evgeny Kiselev

- File *evk_upd20140624a.zip* added

Merged with last revision.

I have still a problem with regression testing. ~ 100 tests have been failed by timeout. Most of tests are failed from every try, but some of tests are failed only in particular tries.

I've tried to remove some of classes from testing and found that problem in refactoring around terminal conversion(Greg update).

How I can determine which tests from result report (for example GSO test) are related to progress source file ?

#38 - 06/24/2014 08:52 PM - Greg Shah

Find a testcase that fails regularly and use the test XML file or the results details to see the recreate. Manually run your server and a client (see the timco.html for details on how to do that). Try to recreate the problem manually. If you can do that, then when you are at the problem location you can use F2 to get to help and the current program can be seen in the PROGRAM item on the help screen.

#39 - 06/24/2014 09:00 PM - Greg Shah

Code Review 0624a

The merge looks fine.

#40 - 07/03/2014 12:02 PM - Evgeny Kiselev

- File *faield2.png* added

- File failed1.png added

I've found the problem.

value = 't'; modifiers = '';
delay = '0'; special = 'NONE';

wait = true; millis = '180000'; passing screen =

06/18/2014 PURCHASE ORDER 564952 TEXT 14:55:0

#	Description
---	-------------

(1)Page Backward (2)Page Forward (A)dd (U)pdate (L)ist (R)return
(3)Half-Page Backward (4)Half-Page Forward: R
Enter data or press F4 to end.

PASSED 0.051

evk@devsrv01: ~/testing/majic/run/client

07/03/2014 PURCHASE ORDER 564952 TEXT 11:48:30

#	Description
---	-------------

(1)Page Backward (2)Page Forward (A)dd (U)pdate (L)ist (R)return
(3)Half-Page Backward (4)Half-Page Forward: R

As you can see on screen shots, there is no border around menus.
One the left side is result without my update, on the right side is result from terminal.
At the moment I'm looking this error. But if you can recommend me where I need to look it will be good.

value = 'a'; modifiers = '';
delay = '0'; special = 'NONE';

wait = true; millis = '180000'; passing screen =

06/18/2014 PURCHASE ORDER 564952 TEXT

#	Description
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	

F1: GO/ACCEPT ESC-?: HELP F9: INSERT MODE F4: UNDO/ABORT

Enter data and press F1 to accept

evk@devsrv01: ~/testing/majic/run/client

(3)Half-Page Backward (4)Half-Page Forward:
'6' WAS NOT A VALID CHOICE - PLEASE TRY AGAIN.
(1)Page Backward (2)Page Forward (A)dd (U)pdate (L)ist (R)return
(3)Half-Page Backward (4)Half-Page Forward:

#	Description
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	

F1: GO/ACCEPT ESC-?: HELP F9: INSERT MODE F4: UNDO/ABORT

#41 - 07/04/2014 02:53 PM - Evgeny Kiselev

After analyzing converted source code, I've found next difference:

without updates code:

```
UnnamedStreams.assignOut(null);
```

with updates code:

```
UnnamedStreams.assignOut(StreamFactory.openTerminalStream());
```

Some of logic in GenericFrame class works different after this changes, for example method:

`com.goldencode.p2j.ui.GenericFrame#display(com.goldencode.p2j.ui.FrameElement[])`

there is code:

```
if (out != null) // doesn't change if out is terminal
{
    display(out, data);
}
else
{
    // set the value in the screen buffer for each widget using the widget
    // id as the index
    copyToScreenBuffer(data);

    // now make the frame visible, bring it to the top and refresh the
    // contents
    view(data);
    refreshWidgets(null);
}
```

in `com.goldencode.p2j.ui.GenericFrame#view(com.goldencode.p2j.ui.FrameElement[])` same problem.

I've tried to add condition `!out.isTerm()` but it not helped, because in `com.goldencode.p2j.ui.chui.ThinClient` class a lot of logic around redirection.

#42 - 07/04/2014 03:40 PM - Evgeny Kiselev

Maybe problem in wrong conversion or unnecessary stream redirection.

I've looked on `text.p` source code and didn't find any streams in this code. But in converted code (either without updates):

```
doBlock(TransactionType.FULL, "blockLabel0", new Block()
{
    public void body()
    {
        if (!_isEqual(EnvironmentOps.getOperatingSystem(), "unix"))
        {
            UnnamedStreams.assignOut(StreamFactory.openFileStream("/dev/null", true, false));
        }
    }
});
```

```

        UnnamedStreams.safeOutput().setEcho(false);
    }
    else
    {
        if (_isEqual(EnvironmentOps.getOperatingSystem(), "msdos"))
        {
            UnnamedStreams.assignOut(StreamFactory.openFileStream("nul", true, false));
        }
    }
}

OnPhrase[] onPhrase0 = new OnPhrase[]
{
    new OnPhrase(Condition.ERROR, Action.RETRY, "gText")
};

doWhile(TransactionType.SUB, "gText", new LogicalExpression(true), onPhrase0, new Block()
{
    public void body()
    {
        OnPhrase[] onPhrase1 = new OnPhrase[]
        {
            new OnPhrase(Condition.ERROR, Action.RETRY, "blockLabel1")
        };

        doBlock(TransactionType.FULL, "blockLabel1", onPhrase1, new Block()
        {
            public void body()
            {
                new FindQuery(xyNotes, (String) null, null, "xyNotes.key asc, xyNotes.seq asc", LockType.NONE).last();

                if (_isUnknown(xyNotes.getKey()))
                {
                    next("gText");
                }
                tLastText.assign(xyNotes.getKey());
                notes.create();
                RecordBuffer.startBatch();
                notes.setKey(plus(tLastText, 1));
                notes.setSeq(new integer(1));
                notes.setFileName(new character("notes-file"));
                notes.setFieldName(new character("notes-field"));
                RecordBuffer.endBatch();
                RecordBuffer.prepare(xyNotes);
                new FindQuery(xyNotes, "xyNotes.id = ?", null, "xyNotes.key asc, xyNotes.seq asc", new Object[]
                {
                    notes.recordID()
                }, LockType.NONE).unique();
                RecordBuffer.prepare(notes);
                new FindQuery(notes, "notes.id = ?", null, "notes.key asc, notes.seq asc", new Object[]
                {
                    xyNotes.recordID()
                }, LockType.EXCLUSIVE).unique();
                leave("gText");
            }
        });
    }
});

UnnamedStreams.assignOut(StreamFactory.openTerminalStream());
// without update here is UnnamedStreams.assignOut(null);

hideMessage(true);

notes.setSeq(new integer(0));
tKey.assign(notes.getKey());
endSeq.assign(0);
}
});

```

#43 - 07/07/2014 02:17 AM - Constantin Asofiei

About note 42: what you are seeing are unnamed input streams (which are opened via a OUTPUT TO ... and INPUT FROM ... statements). As you can notice, there is no explicit stream name used. In P2J, these get converted to UnnamedStreams.assignOut and assignIn API calls.

About note 41: you need to check all the methods which receive a Stream parameter from GenericFrame, and all the UnnamedStreams.output() and UnnamedStreams.input() usage. In cases when you have code like stream != null, you need to replace it with !stream.isTerm(); the other case, where stream == null needs to be replaced with stream.isTerm().

The ThinClient logic should be OK, because this already is implemented with TerminalStream - only the server-side logic was using null to identify a terminal stream (input or output).

#44 - 07/17/2014 03:31 PM - Greg Shah

Please post your latest findings, your latest code and your testing results as they stand.

#45 - 07/17/2014 03:36 PM - Evgeny Kiselev

- File evk_upd20140717a.zip added

Finally I've found a problem. Please review it.

This is dirty fix and not finally, but it have passed regression testing.

I've change ThinClient class because he had his own logic about terminal stream and null stream.

UnnamedStreams was change too to stop use redirect if it terminal stream.

GenericFrame has many changes(maybe I've lost something). In most places I've replaced logic with null stream to TerminalStream

Also all classes was merged with head revision(yesterday).

It was not easy to test it locally, because I couldn't create testcase which exactly replicate situation in gso156_session1 (step 9, see note 40). So I've just added into converted java code UnnamedStreams.assignOut(StreamFactory.openTerminalStream()); in some place.

#46 - 07/17/2014 05:24 PM - Greg Shah

Code Review 0717a

Generally the changes look good to me.

Constantin, what do you think?

Some feedback:

1. In ThinClient.redirectedOutput() there is this code:

```
int old = switchWorker(newId);

// all we need to do is delete it from the map
if (oldId != -1)
{
    Stream out = sd.streamFromId(oldId); <-- this hides the int var named old above

    if (oldId != old && out != null && !out.isTerm()) <-- this compares the int oldId to the Stream old
```

I think this code needs some slightly different naming to work as expected.

2. In ThinClient.switchWorker(), how about changing the code on line 12983 from this:

```
if (newId == -1)
```

to this:

```
if (newId == -1 || sd.streamFromId(newId).isTerm())
```

and then removing the changes to the else block below it.

The only question in my mind is if `redirections.containsKey(key)` would be true when `sd.streamFromId(newId).isTerm()` is true, then the behavior would be different than how you have coded it.

The advantage of this approach is it is less code and it is more clear.

3. Please put `()` around the `out != null && out.isTerm()` change in `UnnamedStreams` (line 498). That will make the precedence more explicit and thus easier to read.

#47 - 07/18/2014 05:48 AM - Constantin Asofiei

Greg Shah wrote:

Constantin, what do you think?

The changes look OK, except for `StreamFactory.openFileStream`, which still needs to treat the case when the filename is `term` or `terminal` - in this case, a `TerminalStream` needs to be opened. This code should be re-added at the start of this method:

```
boolean terminal = false;

if (filename != null)
{
    String term = filename.toLowerCase();

    if (term.equals("terminal") || term.equals("term"))
    {
        term = "terminal";
        terminal = true;
    }
}

// terminal stream without paging is not allowed
if (terminal && !paged && pageSz == NO_PAGING)
    return openTerminalStream();
```

#48 - 07/18/2014 09:01 AM - Greg Shah

I had removed that code when I changed progress.g to match the strings "term" or "terminal" as the KW_TERM. I prefer to keep that "feature" for the conversion and have any non-legacy code changed to explicitly open the terminal stream instead of indirectly doing it through a special name as a fake file.

#49 - 07/18/2014 09:50 AM - Constantin Asofiei

Greg Shah wrote:

I had removed that code when I changed progress.g to match the strings "term" or "terminal" as the KW_TERM. I prefer to keep that "feature" for the conversion and have any non-legacy code changed to explicitly open the terminal stream instead of indirectly doing it through a special name as a fake file.

What about the case when the filename is determined at runtime, i.e.:

```
def stream rpt.  
def var ch as char.  
ch = "term".  
output stream rpt to value(ch).  
display stream rpt ch.  
output stream rpt close.
```

This will still output to terminal, and not to file...

#50 - 07/18/2014 10:13 AM - Greg Shah

Very good point. OK, I guess it is back in.

Evgeny: please make this change and the final changes to finish the update (history entries...). Post that for a final review and it will need one last regression testing run.

#51 - 07/18/2014 06:29 PM - Evgeny Kiselev

ok

#52 - 07/21/2014 02:36 PM - Evgeny Kiselev

- File 01_evk_upd20140721a.zip added

Added missing history entries.
Added Constantine changes from note 47
Fixed issues from note 46

Also I've found other problems with -1 id in ThinClient:
line 2923
line 5050
line 5164
line 9003
line 12614

#53 - 07/21/2014 04:31 PM - Greg Shah

Code Review 0721a

1. In StreamFactory.openFileStream() the call to `int id = work.obtain().rsb.openFileStream(filename, write, append);` must be moved to after the call to:

```
{  
    return openTerminalStream();  
}
```

```
<--- move code here
```

Otherwise, in this "terminal" case an actual file named "term" or "terminal" will be opened.

2. The extra call to `Stream out = sd.streamFromId(newId);` at the top of `ThinClient.switchWorker()` is not needed.

#54 - 07/21/2014 07:12 PM - Evgeny Kiselev

- File `01_evk_upd20140721b.zip` added

fixed issues from note 53

#55 - 07/22/2014 03:20 AM - Constantin Asofiei

Greg, I think we should double-check the `redir*.p` tests from `testcases/uast`. More, the `ThinClient.currentStream` assumes that -1 is always the interactive stream. But, if a terminal stream is opened (with its own ID), then a `currentStream == streamId` test will not produce correct result, when both streams are terminal.

Another question: wouldn't it be better to let the client-side be aware of only one terminal stream? Better said, are we sure it is OK to have multiple client-side terminal streams, each one of them with its own buffer and flush management?

#56 - 07/22/2014 12:09 PM - Greg Shah

wouldn't it be better to let the client-side be aware of only one terminal stream? Better said, are we sure it is OK to have multiple client-side terminal streams, each one of them with its own buffer and flush management?

This is a very good point and I think it is a problem. The 4GL almost certainly only has 1 terminal instance.

One of the most important places where this is an issue is `StreamDaemon.openTerminalStream()` which always creates a new instance. I think we will need to track the open instance when it exists and return that same instance multiple times, until it is no longer valid.

On the server-side a -1 stream id is invalid. We have code in both `StreamFactory` and in `RemoteStream` that uses the concept that valid IDs are always positive. However, I think those are the only dependent places and the logic is not too complex.

`ThinClient.currentStream` assumes that -1 is always the interactive stream. But, if a terminal stream is opened (with its own ID), then a `currentStream == streamId` test will not produce correct result, when both streams are terminal.

You're right. I think there are enough other places where we do depend on that concept that it makes sense to avoid trying to make `ThinClient` safe for the terminal stream ID to be something else.

My idea is to allow -1 to always be used to refer to the current terminal stream, even on the server. If it is not currently open, then we can open a new terminal stream but its ID would always be -1. Most of the `ThinClient` changes can be eliminated in that case, right? Of course, the server side will need changes to handle this, but since we don't expose the IDs out to the business logic, this should be reasonable.

#57 - 07/22/2014 05:45 PM - Evgeny Kiselev

Greg Shah wrote:

My idea is to allow -1 to always be used to refer to the current terminal stream, even on the server. If it is not currently open, then we can open a new terminal stream but its ID would always be -1. Most of the `ThinClient` changes can be eliminated in that case, right? Of course, the server side will need changes to handle this, but since we don't expose the IDs out to the business logic, this should be reasonable.

Do I need to start implement it ?

#58 - 07/23/2014 04:55 AM - Constantin Asofiei

Greg, looking at the logic where `TerminalStream` was previously opened by `StreamDaemon`, I think it was only this case: create a `TerminalStream` instance only if the stream is paged or a page-size is set. See this code (current P2J) in `StreamDaemon`:

```
public int openFileStream(String filename, boolean write, boolean append)
throws ErrorConditionException
{
    if (filename != null && filename.equalsIgnoreCase("terminal"))
        return store(new TerminalStream());
}
```

combined with this code from `StreamFactory.openFileStream`:

```
// terminal stream without paging is not allowed
if (terminal && !paged && pageSz == NO_PAGING)
    return null;
```

Thus, if `StreamDaemon.openFileStream` ends up with a term or terminal filename, then it will store a `TerminalStream`; and this can be possible only if the stream is paged or a page-size is set. I think this shows that P2J (and 4GL) has two flavours of terminal streams:

1. a case when a stream is directed at the terminal. In current P2J, this case is treated via the `TerminalStream`
2. a case when the terminal is accessed directly. In current P2J, this case is treated via -1 by `ThinClient.currentStream`, `GenericFrame.getRemoteStreamId` and maybe others.

The deeper I dig, the more I think that if we merge the `TerminalStream` usage with the direct terminal, we will break something. I can't recall the case why `TerminalStream` was added, but there is too much logic which relies on the fact that an explicit stream is in use or not for it to be a coincidence that we decided to use `TerminalStream` only for the specific case when the stream is paged or page-size is explicitly set.

#59 - 07/23/2014 09:59 AM - Greg Shah

The deeper I dig, the more I think that if we merge the `TerminalStream` usage with the direct terminal, we will break something. I can't recall the case why `TerminalStream` was added, but there is too much logic which relies on the fact that an explicit stream is in use or not for it to be a coincidence that we decided to use `TerminalStream` only for the specific case when the stream is paged or page-size is explicitly set.

Please look back at notes 21 and 22 above. The issue we are dealing with is the case where the terminal has been specifically opened as a named stream. This technique is most often used when the unnamed stream has been redirected to a file and someone came along later needing to output to the screen so they access it as a named stream and intermix the code with the use of the redirected unnamed stream. It is a stupid thing to do, but the 4GL enables it.

As the testcase in those notes demonstrates, the same frame and its current output state can be output to different destinations. How do we resolve this "named stream as a terminal" usage in a way that is clean?

#60 - 07/23/2014 04:43 PM - Constantin Asofiei

Some 4GL background first:

- in 4GL, application code uses statements to write or read data to/from streams
- these streams can be the default stream (we named these the unnamed streams) or a named stream
- the streams can write or read the data to/from certain devices (terminal, file, process).
- initially, the unnamed streams are associated with the terminal device.
- application logic can decide to associate the unnamed streams with another device, while another named stream is associated with the terminal.
- 4GL has special processing when the output stream is paged, especially when the device is the terminal. To implement this specific case, P2J uses the TerminalStream class.

Currently P2J doesn't have default streams - the terminal is accessed directly. More, when the i.e. unnamed output stream is redirected, the terminal's output is automatically sent to that device, and access to the terminal (via other streams) is no longer possible. This can also be read as, instead of allowing multiple streams to access the terminal device, once the unnamed streams are redirected, P2J hides the terminal device and makes it unavailable for other streams. To get past this limitation, P2J uses a switching mechanism - each time an output statement is executed, it checks first if the stmt targets a stream different than the one currently in use; if so, it temporarily switches to that stream, and restores it while work is done. This stream switching feature is generic enough to allow going back to the terminal, even if the unnamed stream is redirected.

Thus: what part from the case in note 21/22 is not working? I've tested with latest P2J and the code looks fine.

#61 - 07/24/2014 09:30 AM - Greg Shah

what part from the case in note 21/22 is not working? I've tested with latest P2J and the code looks fine.

Evgeny: this question is for you.

#62 - 07/24/2014 03:09 PM - Evgeny Kiselev

- File *last_test2.png* added

- File *last_test1.png* added

testcase:

```
def var flag as logical.  
def var displayFrame as logical.  
define stream scr.  
output stream scr to terminal.
```

```

define stream rep.
output stream rep to FieldSizeReport.txt.

displayFrame = yes.

message "display frame" update displayFrame.

display "1)first msg".

output to value ("test.txt") append echo.
display stream scr "2) second msg".
display "3) third msg".
message "test message".
output close.

message "output close" view-as alert-box.

if displayFrame then display stream rep "4) fourth".
display "5) fifth".

message "end" view-as alert-box.

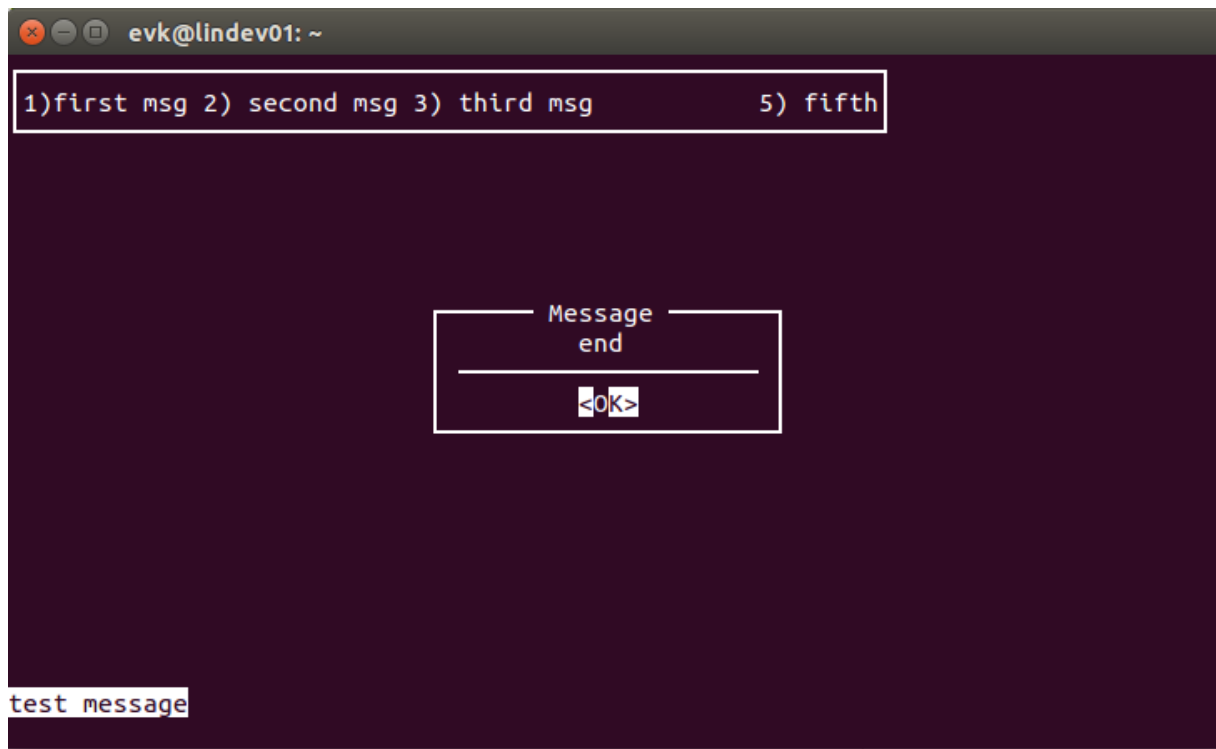
```

run with displayFrame=no

4gl result:



p2j result:



#63 - 07/25/2014 05:30 AM - Constantin Asofiei

I don't think the problem is related to where the output is sent or if the unnamed output stream is redirected or not. Check this case:

```
def var ch1 as char init "1".
def var ch2 as char init "2".
def var i as int.
form ch1 ch2 with frame f1.

message "view frame?" update vf as log.
if vf then view frame f1. /* force frame to be already visible */

message "down in stream?" update isdown as log.

def stream rpt.
output stream rpt to test.txt.
```

```
pause message "ch1".
display stream rpt ch1 with frame f1.
if isdown then down stream rpt with frame f1. /* clear the screen buffer for the frame */

pause message "ch2".
output stream rpt close.

display ch2 with frame f1.

pause message "done".
```

where the final screens are:

- vf = no, isdown = no
4GL:

ch1	ch2
1	2

P2J:

ch1	ch2
1	2

- vf = yes, isdown = no
4GL:

ch1	ch2
1	2

P2J:

ch1	ch2
	2

- vf = no, isdown = yes
4GL:

ch1	ch2
	2

P2J:

ch1	ch2
	2

```
_____
```

- vf = yes, isdown = yes
4GL:

ch1	ch2
2	

P2J:

ch1	ch2
2	

P2J works incorrect in the case when the frame is already visible, and a widget (which is already displayed) has a different value than the one displayed on screen (the internal screen buffer has changed for that widget). This is case (vf = yes, isdown = no) from above. I think what 4GL does is that, when a widget is displayed, it refreshes all the widgets in the frame, with their latest values from the frame's screen buffer. This behaviour looks like is limited to DISPLAY/UPDATE/PROMPT-FOR/ENABLE - VIEW does not exhibit this. This problem is distinct enough to be worked after the main [#1628](#) issue (or as part of another task).

For this task, the next steps are:

- The progress.g change which emits the StreamFactory.openTerminalStream can remain - but conversion rules need to be enhanced to cover all this cases:

```
def stream rpt.  
  
  output stream rpt to term.  
  output stream rpt to term paged.  
  output stream rpt to term paged page-size 10.  
  output stream rpt to term page-size 10.  
  
  output to term.  
  output to term paged.  
  output to term paged page-size 10.  
  output to term page-size 10.  
  
  output stream rpt to term append.  
  output stream rpt to term append paged.  
  output stream rpt to term append paged page-size 10.  
  output stream rpt to term append page-size 10.  
  
  output to term append.  
  output to term append paged.  
  output to term append paged page-size 10.  
  output to term append page-size 10.  
  
  input stream rpt from term.  
  input from term.
```

and to return a null stream if the terminal is not paged and page-size is not set. Note that not all of these convert properly, with the current update - parameters are not properly set and no distinction is made between write and read modes; new StreamFactory.openTerminalStreams APIs need to be added, to accept parameters as the openFileStream API does. Also, the StreamBuilder.openTerminalStream needs to be changed, to accept the write parameter - append I think it can be ignored.

- backout the changes which assume a TerminalStream is in use, when the stream's target device is actually the physical terminal (no paged/page-size set, the old "currentstream=-1" or "stream=nul" cases need to be added back) and:
 - StreamFactory.openFileStream needs to return null if the physical terminal is targeted.
 - StreamFactory.openTerminalStream needs to return null if the physical terminal is targeted (as openFileStream does)
 - StreamDaemon.openFileStream needs to return a TerminalStream instance, if the filename is the term or terminal string.
 - GenericFrame - if the stream is null, then no redirection is in place (the physical terminal is targeted). The !out.isTerm needs to be removed - is enough to assume that if a stream is in place, then the terminal device is not targeted directly
 - the Stream.isTerm API: I think this API is confusing and should be removed. In latest P2J, no class overrides this, so it always returns false.

The only case which was made to return true is the TerminalStream implementation - but this is not the physical terminal either. isTerm should return true only if an actual Stream implementation targets the physical terminal directly - but P2J has no such case.

- ThinClient - same, if stream ID is -1, then the physical terminal is targeted.

#64 - 07/25/2014 09:33 AM - Greg Shah

P2J works incorrect in the case when the frame is already visible, and a widget (which is already displayed) has a different value than the one displayed on screen (the internal screen buffer has changed for that widget). This is case (vf = yes, isdown = no) from above. I think what 4GL does is that, when a widget is displayed, it refreshes all the widgets in the frame, with their latest values from the frame's screen buffer. This behaviour looks like is limited to DISPLAY/UPDATE/PROMPT-FOR/ENABLE - VIEW does not exhibit this. This problem is distinct enough to be worked after the main [#1628](#) issue (or as part of another task).

Good work in isolating the issue. I agree that this should be handled in a separate task. I will create that and we will work it later.

For this task, the next steps are:

...

Evgeny: please follow this plan. Ask questions as needed.

#65 - 07/25/2014 01:27 PM - Evgeny Kiselev

Greg Shah wrote:

Evgeny: please follow this plan. Ask questions as needed.

Just want to clarify:

- For this task I need to rollback changes which contains things related to [#2345](#)?
- Provide update for final review with changes for this task ?

#66 - 07/25/2014 04:08 PM - Greg Shah

For this task I need to rollback changes which contains things related to [#2345](#)?

Yes, following the implementation guidance documented by Constantin in note 63.

Provide update for final review with changes for this task ?

Yes.

#67 - 07/29/2014 01:11 PM - Evgeny Kiselev

- File *evk_upd20140728a.zip* added

Fixed everything from note 63 except conversion rules.

#68 - 07/29/2014 01:55 PM - Greg Shah

Code Review 0728a

I am fine with the changes.

#69 - 07/31/2014 09:16 AM - Constantin Asofiei

Evgeny: please merge 0728a.zip with the latest bzip revision - #2352 has changes which need to be merged.

#70 - 07/31/2014 06:10 PM - Evgeny Kiselev

- File *evk_upd20140731a.zip* added

final version with properly conversion
merged with latest bzip revision

#71 - 08/01/2014 03:54 AM - Constantin Asofiei

Evgeny, about 0731a.zip: the `StreamFactory.openTerminalStream(boolean, int, boolean)` needs to return `NULL` only if the stream is not paged and its page-size is not set, i.e. if `(!paged && pageSz == NO_PAGING) { return null; }`. In all other cases, (stream paged or page size is set), a real `TerminalStream` needs to be constructed.

More, the conversion of the `openTerminalStream` APIs doesn't seem correct. Check this case: output to term. is converted to `openTerminalStream(false)`, but the signature of this API is `boolean paged` - I think the correct signature of this API is `boolean append`.

After you make these changes, go ahead and get it conversion and runtime tested.

#72 - 08/01/2014 12:32 PM - Evgeny Kiselev

- File *evk_upd20140801a.zip* added

Constantin Asofiei wrote:

Evgeny, about 0731a.zip: the `StreamFactory.openTerminalStream(boolean, int, boolean)` needs to return `NULL` only if the stream is not paged and its page-size is not set, i.e. if `(!paged && pageSz == NO_PAGING) { return null; }`. In all other cases, (stream paged or page size is set), a real `TerminalStream` needs to be constructed.

Do I need to specify in `com.goldencode.p2j.util.StreamDaemon#openTerminalStream` page size ?
Where is a method `rawSetPageSize` for it.

#73 - 08/01/2014 12:49 PM - Constantin Asotiei

Evgeny Kiselev wrote:

Do I need to specify in `com.goldencode.p2j.util.StreamDaemon#openTerminalStream` page size ?

No, I don't think so. In the previous logic, `StreamDaemon.openFileStream` doesn't pass the page/page-size to the new `TerminalStream` instance.
The changes look good. Go ahead and regression test them (conversion and runtime).

#74 - 08/04/2014 07:15 PM - Evgeny Kiselev

- File `evk_upd20140803a.zip` added

merged with latest revision (`ThinClient` class)

Regression testing failed with 10 tests in main part and 3 tests in `ctrlc` part.
Some of the tests were not appear in different runs, but other tests is appear repeatably:
1) one tests failed with: `** Attempt to write to closed stream to-screen. (1387)`
2) in two test was error with incorrect amount of pages: `Record(s) selected: 653 22 Pages (but should be different value)`
Continue debugging.

#75 - 08/08/2014 05:51 AM - Evgeny Kiselev

- File `01_evk_upd20140805a.zip` added

`ctrlc` part passed
main part failed, but problem was in testsuit, now tests are running again.

different from previous update:
`GenericFrame` line 7491:
in old code was: `redirOut != null && redirOut.isTerm();` but this code never have been true, because `redirOut.isTerm()` was always false. I'm not sure about correctness of this part.
`StreamWrapper` line 404:

before it was always true inOk = true; and outOk = true;, maybe more correct:
inOk = stream == null || stream.isIn(); and outOk = stream == null || stream.isOut();

#76 - 08/08/2014 10:24 AM - Greg Shah

Code Review 0805a

I agree that an alternate approach in StreamWrapper seems more correct.

StreamWrapper line 404:

before it was always true inOk = true; and outOk = true;, maybe more correct:
inOk = stream == null || stream.isIn(); and outOk = stream == null || stream.isOut();

If you make this change, please format it like this:

```
inOk  = (stream == null || stream.isIn());  
outOk = (stream == null || stream.isOut());
```

Have you tested this approach?

#77 - 08/08/2014 10:26 AM - Greg Shah

I edited the previous note to fix the logic I had broken...

#78 - 08/11/2014 09:39 AM - Evgeny Kiselev

In StreamWrapper with next code:

```
inOk  = (stream == null || stream.isIn());  
outOk = (stream == null || stream.isOut());
```

Tests are failed. The main issue is "*** Attempt to write to closed stream rpt. (1387)". As I understand there are a lot of places where something writes in close stream.

With:

```
inOk  = true;  
outOk = true;
```

Tests are failed too, but just few of them and didn't repeat too often. And also ctrlc part failed.

#79 - 08/11/2014 09:48 AM - Greg Shah

OK, then we need to leave the code as:

```
inOk  = true;
outOk = true;
```

Please document (in comments inside that method) that you have tried the alternate approach and how it failed. Then post the final code here.

Tests are failed too, but just few of them and didn't repeat too often. And also ctrlc part failed.

We need to get this through testing. Are there any tests which are always failing?

#80 - 08/11/2014 09:51 AM - Evgeny Kiselev

Greg Shah wrote:

We need to get this through testing. Are there any tests which are always failing?

CTRL-C part:

ctrlc_11_session1 step 18-28
ctrlc_11_session3 step 9-23, 25
ctrlc_11_session4 step 1-26, 28

Main part:

gso_430
gso_276
tc_job_002
tc_job_clock_002

#81 - 08/11/2014 10:28 AM - Greg Shah

tc_job_002
tc_job_clock_002

Are these the known failures as described in our documentation?

The other problems definitely need to be investigated. Please provide more details of your findings.

#82 - 08/13/2014 05:55 AM - Evgeny Kiselev

CTRL-C part:
ctrlc_11_session1 step 18-28
ctrlc_11_session3 step 9-23, 25
ctrlc_11_session4 step 1-26, 28

Failed with error:
Timeout while waiting for event semaphore to be posted.

Two tests failed with:
Timeout before the specific screen buffer became available (Mismatched data at line 1, column 0. Expected ' ' (0x250C at relative Y 1, relative X 0) and found 'e' (0x0065 at relative Y 1, relative X 0).)

Differents:
Should be " ┌──────────" (top left corner border of window), but found:

```
Killed2014evk@devsrv01:~$ ^Cion: TIPR833K  TIMCO - GREENSBORO  
evk@devsrv01:
```

I've tried to run this tests manually and they worked fine.

main part:
gso_276 step 33
Wrong result Record(s) selected: 167 25 Pages, but should be Record(s) selected: 167 24 Pages (checked manually confirm this issue)

Next tests were failed time to time:

gso_269	step 48
gso_430	step 19
gso_290	step 19
gso_167	step 35
gso_293	step 38

#83 - 08/13/2014 06:02 AM - Constantin Asofiei

Evgeny Kiselev wrote:

CTRL-C part:
ctrlc_11_session1 step 18-28
ctrlc_11_session3 step 9-23, 25
ctrlc_11_session4 step 1-26, 28

Failed with error:

These are not of concern, they are usual "false negatives".

main part:
gso_276 step 33
Wrong result Record(s) selected: 167 25 Pages, but should be Record(s) selected: 167 24 Pages (checked manually confirm this issue)

Next tests were failed time to time:
gso_269 step 48
gso_430 step 19
gso_290 step 19
gso_167 step 35
gso_293 step 38

Did this pass in a previous run?

#84 - 08/13/2014 06:07 AM - Evgeny Kiselev

Also When we removed isTerm() method, there are next possible issues:
RemoteStream had constructor RemoteStream(int id, boolean term) and term = true could be only in one situation in
com.goldencode.p2j.util.StreamFactory.openFileStream(java.lang.String, boolean, boolean, int, boolean) when filename is "terminal" and stream is
paged or page size specified.

Also in com.goldencode.p2j.ui.GenericFrame.isOutTerm() could be true only when stream.isTerm() = true, after refactoring we've lose this
functionality.

Same problem in `com.goldencode.p2j.util.KeyReader.readKey(com.goldencode.p2j.util.Stream, double)` we've lose some functionality.

#85 - 08/13/2014 06:17 AM - Constantin Asofiei

Evgeny Kiselev wrote:

Also When we removed `isTerm()` method, there are next possible issues:

`RemoteStream` had constructor `RemoteStream(int id, boolean term)` and `term = true` could be only in one situation in `com.goldencode.p2j.util.StreamFactory.openFileStream(java.lang.String, boolean, boolean, int, boolean)` when filename is "terminal" and stream is paged or page size specified.

You are correct, `isTerm` is overridden by `RemoteStream` and code on server-side is dependent on this. Please add back these.

#86 - 08/15/2014 04:50 AM - Evgeny Kiselev

- File `01_evk_upd20140812a.zip` added

update:

- 1) rollback `isTerm()` method
- 2) rollback `ThinClient`(described below)
- 3) I've added in `StreamFactory` next code:

```
if (terminal) // instead of if (terminal && !paged && pageSz == NO_PAGING)
{
    return openTerminalStream();
}
```

I'm not sure about this part of code, because if it's terminal, we need to open real terminal stream (if paged and `pageSz` are specified) or open file stream if filename is not "term" or "terminal"

`ThinClient`:

I've had rollback this class, because after some rollbacks described upper, this code became works incorrect(in latest revision same problem), here is two testcases:

```
output to value ("test.txt") append.
```

```
message "message:test1".
display "some test,".
message "message:test2".
put unformatted "this is the appended text".
```

```
output close.
message "output close" view-as alert-box.
```

Every testcase I've run two times (because of append flag)

Results:

4gl:

```
message:test1

some test,
message:test2
this is the appended textmessage:test1

some test,
```



```
message:test2
this is the appended text
```

p2j:

```
message:test1

some test,
    message:test2
this is the appended textmessage:test1

some test,
    message:test2
this is the appended text
```

This current (latest revision) code. My fix for ThinClient could resolve problem only for first message (in test in #1629 I had only one message after display), but it doesn't work after redirection is already set. That's why it produced wrong result for gso_276 testcase.

```
output to value ("test.txt") append.

put unformatted "~n ".
message "message:test1".
display "some test,".
message "message:test2".
put unformatted "this is the appended text".

output close.
message "output close" view-as alert-box.
```

Results:

4gl:

```
message:test1

some test,
message:test2
this is the appended text

message:test1

some test,
message:test2
this is the appended text
```

p2j:

```
message:test1

some test,
    message:test2
this is the appended text  message:test1

some test,
    message:test2
this is the appended text
```

This testcase gives interesting results. As we can see a gap for first message:test1 for two symbols (\n and space, from put unformatted) but we didn't see this symbols in file ! Also if we remove first message from code, we will see in file this symbols, but we will add two extra spaces for message2 line.

I've tried to find solution in RedirectedTerminal but didn't fine any suitable fix.

This update passed gso_276 tests, now it's running full-regression

Code Review 0812a

The proposed code can be accepted with only some minor changes (listed below).

However, I think the problems 10/11 in #1629 note 17 will NOT be fixed since you have undone the changes in GenericFrame and ThinClient. Am I correct? These problems are related to NO-ECHO support, so they do need to get fixed. I would be OK with getting the current changes through testing and checked in, before fixing the last problems. But we must fix those before we can close out this task.

In regard to the two samples listed in note 86: does the P2J behavior match the 4GL with your 0812a update? In other words, these problems are with the changes needed to fix the problems 10/11 in #1629 note 17?

Cleanup for 0812a:

1. At this point the StreamWrapper changes are only in formatting and comments. The history entries are no longer valid. Please just change my history entry to "Minor cleanup of formatting and comments." and remove your history entry.
2. The removal of `&& !paged && pageSz == NO_PAGING` from StreamFactory seems dangerous. Please provide more information on why you made this change. I also wonder if part of our problem is that this code:

```
if (term.equals("terminal") || term.equals("term"))
{
    term = "terminal";
    terminal = true;
}
```

should probably be this:

```
if (term.equals("terminal") || term.equals("term"))
{
    filename = "terminal";    <--- this code was setting term but it was supposed to set filename
    terminal = true;
}
```

#88 - 08/22/2014 09:32 AM - Evgeny Kiselev

problem described in note 86 divide into 2 problems:

1) incorrect position for message method with redirected stream (I think I've fixed this problem, now tests are running)

2) problem with put unformatted "~n ":

at the moment this code converted into:

```
UnnamedStreams.safeOutput().putUnformatted(new FieldEntry[]
{
    new PutField("\n ")
});
```

and in method `com.goldencode.p2j.util.Stream#generateFormattedText` applies format for this string. And string becomes " "(just two spaces, instead of \n and space). Also 4gl added new line after if next output is formatted message or display.

#89 - 08/25/2014 09:24 AM - Greg Shah

problem with put unformatted "~n ":

...

and in method `com.goldencode.p2j.util.Stream#generateFormattedText` applies format for this string

This doesn't seem correct. How does `Stream.putUnformatted(FieldEntry[])` call `Stream.generateFormattedText()`? `putUnformatted()` calls `putWorker()` and `generateFormattedText()` is only called by two of the `putField()` methods.

#90 - 08/25/2014 10:00 AM - Evgeny Kiselev

Here is my stacktrace:

```
main@1, prio=5, in group 'main', status: 'RUNNING'
  at com.goldencode.p2j.util.Stream.generateFormattedText (Stream.java:5984)
  at com.goldencode.p2j.util.Stream.putField (Stream.java:3437)
  at com.goldencode.p2j.util.PutField.output (PutField.java:946)
  at com.goldencode.p2j.util.Stream.putWorker (Stream.java:5176)
  at com.goldencode.p2j.util.StreamDaemon.putWorker (StreamDaemon.java:1005)
```

#91 - 08/25/2014 10:48 AM - Greg Shah

OK, that makes more sense now. What do you suggest?

#92 - 08/26/2014 08:35 AM - Evgeny Kiselev

- File 01_evk_upd20140820a.zip added

fix problem described in note 86
fix issues from note 87
merged with latest revision

Before merge this update have been passed regression testing, but now it not compile because of ChuiScreenDriver class.

#93 - 08/26/2014 09:45 AM - Constantin Asofiei

Evgeny Kiselev wrote:

Before merge this update have been passed regression testing, but now it not compile because of ChuiScreenDriver class.

It was a problem in rev 10602, 10603 is OK now.

#94 - 08/26/2014 10:34 AM - Greg Shah

Code Review 0820a

I am fine with the changes.

However, I think the problems 10/11 in #1629 note 17 will NOT be fixed since you have undone the changes in GenericFrame and ThinClient. Am I correct? These problems are related to NO-ECHO support, so they do need to get fixed. I would be OK with getting the current changes through testing and checked in, before fixing the last problems. But we must fix those before we can close out this task.

I still don't know the answer to this question from note 87.

And I would like to know your suggested approach to solving the PUT UNFORMATTED newline issue.

Are there any other open issues to resolve?

#95 - 08/26/2014 01:32 PM - Evgeny Kiselev

Greg Shah wrote:

I still don't know the answer to this question from note 87.

And I would like to know your suggested approach to solving the PUT UNFORMATTED newline issue.

Are there any other open issues to resolve?

Yes, this is last one issue.

#96 - 08/26/2014 02:04 PM - Evgeny Kiselev

I know, that 4gl documentation is not always correct, but there is next paragraph in put statement:

UNFORMATTED
Tells the AVM to display each expression in the same format produced by the
EXPORT statement, but without quotes.

In our code, we add extra new line only for MODE_EXPORT, but in this situation we need to add extra line only when we have some other output. Also if I fix problem with formatting, it didn't calculate \n symbols while calculating number of characters in particular line.

#97 - 08/26/2014 02:52 PM - Greg Shah

In our code, we add extra new line only for MODE_EXPORT, but in this situation we need to add extra line only when we have some other output.

Are you referring to the use of endOutput(true) in Stream.putWorker()?

Also if I fix problem with formatting,

What is your proposed formatting fix?

it didn't calculate \n symbols while calculating number of characters in particular line.

What code are you referring to here?

#98 - 09/10/2014 07:44 AM - Evgeny Kiselev

- File 01_evk_upd20140910a.zip added

I've added addition comments in code. There is next algorithm.

1) first output is put unformatted (in my case this is "\n "). I've disabled formatting in com.goldencode.p2j.util.Stream#generateFormattedText if we are in unformatted mode.

4gl will output:

```
NEW_LINE
WHITESPACE
```

if next put unformatted will be added, 4gl will continue output from last position.

2) next output is message: 4gl will add additional new line to the last known position. I've added lazyTrailingNL variable in Stream class. New Line will be added only if next output is message or display. If put unformatted is the last output, new line will not be added.

3) in ThinClient:2987 I've changed cursor position for all messages to 0 on X-axis.

Also in class RedirectedTerminal I've added additional code to reduce one space shift, because after first display box[i] array become always true.

#99 - 09/10/2014 03:01 PM - Greg Shah

Code Review 0910a

I am worried that your changes are just there to fix some very specific behavior in a few testcases, but that the fixes are not general purpose. I expect these changes to cause serious problems.

1. Why is lazyTrailingNL only set true in putField(BaseDataType val, String fmt)? There are many other paths through variants of putField() that will bypass this. This doesn't seem right. Also, if you do call writeNewLine(), then we may also need to call rawIncrementNextLineNum(). But more importantly, I don't see why adding this lazyTrailingNL is correct at all. What you are describing is just that PUT UNFORMATTED should output its contents without any consideration of newline processing, right? Perhaps we just have PUT UNFORMATTED implemented wrong.

2. Your TerminalStream version is not merged with the latest version in bsr.

3. The RedirectedTerminal change seems dangerous and incorrect. box[i] will only be true when the BasePrimitives.box() method is called. This can happen for any frame that is output where it is NOT configured with the NO-BOX option. Of course, for most frames it is true, but I still don't see why you are making the change. In addition, the redirected output to streams doesn't change for file streams versus other streams, as far as I know. For example:

```
output through "cat > bogus.txt".
display "hello".
output close.
```

and

```
output to "bogus.txt".
display "hello".
output close.
```

Both of these generate a bogus.txt file that looks like this:

```
hello
```

4. You are missing history entries and some places have { that should start on the next line.

#100 - 09/10/2014 03:06 PM - Greg Shah

What is the status of your regression testing?

What standalone testcases have you run to confirm your changes are correct? I think you need to run the redirected_*.p tests in uast/testcases/.

#101 - 09/10/2014 07:04 PM - Evgeny Kiselev

Greg Shah wrote:

Code Review 0910a

I am worried that your changes are just there to fix some very specific behavior in a few testcases, but that the fixes are not general purpose. I expect these changes to cause serious problems.

1. Why is lazyTrailingNL only set true in putField(BaseDataType val, String fmt)? There are many other paths through variants of putField() that will bypass this. This doesn't seem right. Also, if you do call writeNewLine(), then we may also need to call rawIncrementNextLineNum(). But more importantly, I don't see why adding this lazyTrailingNL is correct at all. What you are describing is just that PUT UNFORMATTED should output its contents without any consideration of newline processing, right? Perhaps we just have PUT UNFORMATTED implemented wrong.

No, this is two different problems

1)

What you are describing is just that PUT UNFORMATTED should output its contents without any consideration of newline processing, right?

yes, this is first problem, but easy to fix.

2) I will try to explain second problem in the next examples.

```
output to value ("test1.txt") append.  
put unformatted "unformat". /*just common string without any special symbols*/  
message "message:test".  
put unformatted "this is the appended text".  
output close.
```

4gl output is:

```
unformat  
message:test  
this is the appended text
```

p2j output is:

```
unformatmessage:test  
this is the appended text
```

I'm trying to explain that 4gl adds new line after each put unformatted if next output is message or display (or maybe good to say that new line added before message or display if previous output was put unformatted). But this new line is not added just after put, it added only before next output. For example last line didn't have new line. Also between two or more put unformatted new line was not added. That's why I've added lazyTrailingNL

3. The RedirectedTerminal change seems dangerous and incorrect. box[i] will only be true when the BasePrimitives.box() method is called.

This can happen for any frame that is output where it is NOT configured with the NO-BOX option. Of course, for most frames it is true, but I still don't see why you are making the change. In addition, the redirected output to streams doesn't change for file streams versus other streams, as

far as I know. For example:

In my testcase box[] array has been filled just after first display method occur and next message become shifted in 1 space. I'll try to find better decision.

#102 - 09/10/2014 07:08 PM - Evgeny Kiselev

Greg Shah wrote:

What is the status of your regression testing?

I've just started regression testing with latest update.

What standalone testcases have you run to confirm your changes are correct?

yes.

I think you need to run the redirected_*.p tests in uast/testcases/.

Ok

#103 - 09/11/2014 07:23 AM - Constantin Asofiei

Evgeny Kiselev wrote:

What standalone testcases have you run to confirm your changes are correct?

I think Greg was asking the name of the testcases you used to discover/fix/test your changes. Please commit these to bzt, so I can understand your reasoning/findings behind your changes.

#104 - 09/11/2014 12:16 PM - Evgeny Kiselev

committed testcases into testcases/uast/no_echo
revision 1199.

testcases:

uast/no_echo/put_unformatted_bug.p
uast/no_echo/put_unformtted_bug1.p

#105 - 09/11/2014 02:07 PM - Constantin Asofiei

Evgeny Kiselev wrote:

testcases:

uast/no_echo/put_unformatted_bug.p
uast/no_echo/put_unformtted_bug1.p

1. please check these issues with a normal PUT. I think what you found applies to it too.
2. there is the SKIP clause for the PUT statement, which allows you to advance to the next row. Check what happens if the PUT ends with a SKIP or SKIP(1).
3. about MESSAGE (and maybe other statements, like DISPLAY) when output is not the terminal: a possibility can be that these statements are checking the cursor position, and automatically start a new line if the cursor is not on the very first column. I think this is confirmed by the following test, which uses SEEK to reposition to a previous position in the file:

```
output to test1.txt.  
put "this is a very long text" skip  
    "this is line 2 of this text" skip  
    "this is line 3 of this text".  
seek output to 4.  
message "this message is on a new line".  
output close.
```

the output is this:

```
this  
this message is on a new line  
ne 2 of this text  
this is line 3 of this text
```

without SEEK output is this:

```
this is a very long text  
this is line 2 of this text  
this is line 3 of this text  
this message is on a new line
```

4. about the box issue:

- is this a regression from your update or a bug you found?
- if is a bug you found, please confirm the bug exists with rev 10588 (before the ScreenBitmap changes were introduced)
- do you have a specific test scenario to duplicate this issue?

#106 - 09/15/2014 04:39 AM - Evgeny Kiselev

Constantin Asofiei wrote:

1. about the box issue:
 - is this a regression from your update or a bug you found?
 - if is a bug you found, please confirm the bug exists with rev 10588 (before the ScreenBitmap changes were introduced)

Yes this bug exists in rev 10588

#107 - 09/15/2014 01:06 PM - Evgeny Kiselev

Constantin Asofiei wrote:

1. about MESSAGE (and maybe other statements, like DISPLAY) when output is not the terminal: a possibility can be that these statements are checking the cursor position, and automatically start a new line if the cursor is not on the very first column. I think this is confirmed by the following test, which uses SEEK to reposition to a previous position in the file:

```
output to test1.txt.  
put "this is a very long text" skip  
    "this is line 2 of this text" skip  
    "this is line 3 of this text".  
seek output to 4.  
message "this message is on a new line".  
output close.
```

the output is this:

```
this  
this message is on a new line  
ne 2 of this text  
this is line 3 of this text
```

P2J output:

```
this  
this message is on a new line  
ine 2 of this text  
this is line 3 of this text
```

P2J output with my patch:

```
this  
this message is on a new line  
ne 2 of this text  
this is line 3 of this text
```

without SEEK output is this:

```
this is a very long text
this is line 2 of this text
this is line 3 of this text
this message is on a new line
```

P2J output:

```
this is a very long text
this is line 2 of this text
this is line 3 of this textthis message is on a new line
```

P2J output with my patch:

```
this is a very long text
this is line 2 of this text
this is line 3 of this text
this message is on a new line
```

#108 - 09/15/2014 04:12 PM - Constantin Asofiei

Evgeny Kiselev wrote:

P2J output with my patch:

If you are referring to 0910a.zip, try to break it like this:

```
output to test1.txt.
put "this is a very long text" skip
  "this is line 2 of this text" skip
  "this is line 3 of this text".
message "something bogus". /* just so last stmt is not PUT */
seek output to 4.
message "this message is on a new line".
output close.
```

If the last stmt which wrote to the stream is not PUT, then I expect your 0910a.zip to fail.

#109 - 09/15/2014 05:57 PM - Evgeny Kiselev

Constantin Asofiei wrote:

Evgeny Kiselev wrote:

P2J output with my patch:

If you are referring to 0910a.zip, try to break it like this:

[...]

If the last stmt which wrote to the stream is not PUT, then I expect your 0910a.zip to fail.

This testcase is working too for 0910a.zip:

4gl result:

```
this this message is on a new line
ine 2 of this text
this is line 3 of this text
something bogus
```

P2J latest revision:

```
this this message is on a new line
ine 2 of this text
this is line 3 of this text something bogus
```

P2J with 0910a.zip

```
this this message is on a new line
ine 2 of this text
this is line 3 of this text
something bogus
```

Evgeny Kiselev wrote:

3. The RedirectedTerminal change seems dangerous and incorrect. `box[i]` will only be true when the `BasePrimitives.box()` method is called. This can happen for any frame that is output where it is NOT configured with the NO-BOX option. Of course, for most frames it is true, but I still don't see why you are making the change. In addition, the redirected output to streams doesn't change for file streams versus other streams, as far as I know. For example:

In my testcase `box[]` array has been filled just after first display method occur and next message become shifted in 1 space. I'll try to find better decision.

The box problem is in fact related to something else: the frame's height (in redirected mode) is computed wrong. It adds a unit for the top border (which is OK) and another unit for the bottom border; and the MESSAGE is actually displayed on this "bottom border" line (thus `box[i]` is true). But I don't think 4GL adds a new-line after a displayed frame; this case:

```
def var ch as char.  
  
ch = "this is a test".  
  
form ch with frame fl.  
  
output to a.txt.  
display ch with frame fl.  
message "this is a message".  
output close.
```

will produce this:

```
ch  
-----  
this is
```

with no empty line after the frame.

Greg: do you think that adding a row at the bottom of the frame (when output to stream) might be something specific to 4GL 9.1? Or is a defect in P2J?

#111 - 09/17/2014 08:17 AM - Constantin Asofiei

- File `ca_upd20140917a.zip` added

Evgeny, this version is merged with the latest P2J. What else needs to be done:

1. test all PUT versions (PUT ... AT/TO, PUT ... CONTROL, SKIP/SPACE clauses), as the `Stream.lazyTrailingNL` usage is incomplete
2. the issue at note 110. I wonder if the lazy trailing for PUT needs to be applied for DISPLAY (and maybe other cases, like EXPORT), too...
another question is if current behaviour is not specific to 4GL 9.1c.

#112 - 09/17/2014 08:54 AM - Greg Shah

do you think that adding a row at the bottom of the frame (when output to stream) might be something specific to 4GL 9.1?

It is possible but I think it is very unlikely. Such a change would potentially break many (most?) working applications that depend upon the redirected terminal processing for their reports.

Or is a defect in P2J?

This is much more likely.

#113 - 09/18/2014 09:22 AM - Evgeny Kiselev

Constantin Asofiei wrote:

update 0917

for `TerminalStream` class you've rollback next lines:

```
worker.cursorAt(0, 0);  
worker.sync();
```

This lines was deleted in 10602 revision. Does these lines needed ?

#114 - 09/18/2014 12:49 PM - Constantin Asofiei

Evgeny Kiselev wrote:

Constantin Asofiei wrote:

update 0917

for TerminalStream class you've rollback next lines:

[...]

This lines was deleted in 10602 revision. Does these lines needed ?

Please remove them, this was a mistake from my part - I thought these were from your code.

#115 - 09/22/2014 08:54 AM - Evgeny Kiselev

I've found problem in calculating border for frame.

Border is calculating correct, but message will appear on the "border", for example (testcase from not 110):

1) we have the frame:

```
ch
-----
this is
```

size of this frame is 3x8, then we've added 1 border around = 5x10. Everything is calculated properly and box[] arrays filled from 0 to 4 to true. But first message just after frame in 4gl will be printed on the bottom border

	0 row	top border
ch	1 row	
-----	2 row	
this is	3 row	
this is a message	4 row	bottom border

maybe we need ignore last row of border for all messages ?

#116 - 09/23/2014 09:02 AM - Greg Shah

In the redirected terminal output (see `RedirectedTerminal.sync()`), we are not supposed to generate output when:

```
// output a new line for every touched line except for a
// touched last line that has no data and had box drawing
```

In other words, the bottom border normally does not get output when a frame is drawn with BOX on the redirected terminal.

The message statement just outputs at the current row in the redirected terminal. You have already found that we needed to reset the X coordinate to the 0 position. That makes sense.

It seems to me that the "current row" may be wrong when we draw. It may be below the border instead of on the same line as the border.

#117 - 09/30/2014 08:56 AM - Greg Shah

What is the status of your work on this task?

#118 - 10/06/2014 08:26 PM - Evgeny Kiselev

- File `01_evk_upd20141003a.zip` added

merged with latest revision

fixed issue with `box[]`

In `Stream lazyNewLine` now working only with `put` operation, `EXPORT` should not add lazy new line

#119 - 10/07/2014 09:07 AM - Greg Shah

Code Review `evk_upd20141003a.zip`

1. The redirected terminal change now looks much safer.

2. Please describe the testing that you did that proves that only the `putField(BaseDataType, String)` should have `lazyTrailingNL` enabled. There are many other `putField()`, `put()`, `putControl()`, `putNull()`, `putLineEnd()`, `putSpace()` and `putUnformatted()` methods which do not flow through this path. Should they be protected? I suspect most or all of these should be protected. If you need to protect all PUT calls, then you should find the workers:

- `putWorker()` handles all output for `put()`, `putControl()` and `putUnformatted()` variants
- `putField(BaseDataType, String, boolean, int)` handles all the other `putField()` variants
- `putSpace(int)` handles `putSpace()` too
- `putNull(int)` handles `putNull()` too
- `putLineEnd(int)` handles `putLineEnd()` too

So there are 5 possible places you might have to change to cover all possible PUT variations. Please provide specific details on how you tested each of these cases and what protection they do or do not need.

3. Please describe the testing you have done to check if `DISPLAY`, `VIEW`, `MESSAGE` and similar redirected terminal usage need the `lazyTrailingNL` support.

4. Please check in your testcases for the PUT and EXPORT `lazyTrailingNL` testing.

5. Please look at how `writeNewLine()` is used in `endOutput()` and `processLineEnd()`. When you use it, don't you also need to call `rawIncrementNextLineNum()`?

6. In `StreamFactory` and `StreamDaemon`, please merge the 2 history entries (one by me and one by you) into a single history entry. You can put your initials there.

7. Did you test the uast/testcases/redirected_*.p tests as I requested in note 100? What are the results of that testing?

I EXPECT YOU TO RESPOND IN DETAIL TO ALL MY QUESTIONS ABOVE. Several of these questions I have noted weeks and months ago, but you have never responded.

#120 - 11/20/2014 09:10 AM - Greg Shah

From Evgeny:

----- Original Message -----

Subject: Re: Status Report

Date: Thu, 20 Nov 2014 02:28:03 +0300

From: Evgeny Kiselev <evk@goldencode.com>

To: ges@goldencode.com

CC: Faulhaber, Eric <ecf@goldencode.com>, evgeni.kiselev@gmail.com

Hello Greg,

I still working on answers for note 119 from [#1628](#). Most of question I have answer and as soon as I'll have second question I'll post results. But then I rerun tests from uast/testcases/redirected_*.p, some of tests became not working because of conversion:

redirected_report6_3.p is not converted properly
redirected_report6_4.p is not converted properly
redirected_report6_5.p is not converted properly
redirected_report6_6.p is not converted properly
redirected_report6_7.p is not converted properly

I remembered they were work properly recently. I think it cause then widget ids became dynamic.

Also I received Nullpointer (I think it's the same problem):

Caused by: java.lang.NullPointerException
at com.goldencode.p2j.ui.client.WindowManager.findFocus(WindowManager.java:132)
at com.goldencode.p2j.ui.client.event.EventManager.eventFromKey(EventManager.java:332)
at com.goldencode.p2j.ui.client.TypeAhead.dequeueEvent(TypeAhead.java:419)
at com.goldencode.p2j.ui.client.TypeAhead.getKeystroke(TypeAhead.java:312)
at com.goldencode.p2j.ui.chui.ThinClient.waitForEvent(ThinClient.java:11107)
at com.goldencode.p2j.ui.chui.ThinClient.pause(ThinClient.java:6467)
at com.goldencode.p2j.ui.chui.ThinClient.pause(ThinClient.java:6364)
at com.goldencode.p2j.ui.chui.ThinClient.pauseBeforeHide(ThinClient.java:2838)
at com.goldencode.p2j.ui.client.Window.message(Window.java:1880)
at com.goldencode.p2j.ui.client.Window.message(Window.java:1972)
at com.goldencode.p2j.ui.chui.ThinClient.message(ThinClient.java:6026)

This problems are on latest unmodified revision.

On 11/18/2014 05:27 PM, Greg Shah wrote:

Evgeny,

I'am preparing answers for note 119 from [#1628](#). Am right ?

Yes.

Thanks,
Greg

On 11/17/2014 10:05 PM, Evgeny Kiselev wrote:

Hello,

I believe that we talk about [#1628](#).

At the moment I've merged with latest code, and retesting.
I have answers for most of the questions, but didn't have prove for some of them.

I'am preparing answers for note 119 from [#1628](#). Am right ?

Regards,
Evgeny.

#121 - 11/20/2014 09:15 AM - Greg Shah

Please provide more information about the conversion problem(s) that occurred.

I would be surprised if the conversion problem was the same as the NPE caused by `WindowManager.findFocus()`. The NPE is in code that is only executed at runtime. It should not come into play during conversion.

For the NPE, `WindowManager.topWindow()` must return null which means that no window has ever been added to the window list. This occurs in `Window.pushConfig()`, `Window.getDefaultWindow()` and in `TitledWindow.show()`. This means that you have found a path through our code where the default window initialization has not occurred. Please post the 4GL code here (or point to the testcase/uast/ file) that can be used to recreate the problem.

#122 - 12/17/2014 07:01 PM - Evgeny Kiselev

- File `evk_upd20141216a.zip` added

Merged with latest revision.

Moved some logic from `ThinClient` to `Window` class

Fix problem in `RedirectedTerminal` covered by `redirected_boxed_frame.p` testcase.

Greg Shah wrote:

2. Please describe the testing that you did that proves that only the `putField(BaseDataType, String)` should have `lazyTrailingNL` enabled. There are many other `putField()`, `put()`, `putControl()`, `putNull()`, `putLineEnd()`, `putSpace()` and `putUnformatted()` methods which do not flow through this path. Should they be protected? I suspect most or all of these should be protected. If you need to protect all PUT calls, then you should find the workers:

- `putWorker()` handles all output for `put()`, `putControl()` and `putUnformatted()` variants
- `putField(BaseDataType, String, boolean, int)` handles all the other `putField()` variants
- `putSpace(int)` handles `putSpace()` too
- `putNull(int)` handles `putNull()` too
- `putLineEnd(int)` handles `putLineEnd()` too

So there are 5 possible places you might have to change to cover all possible PUT variations. Please provide specific details on how you tested each of these cases and what protection they do or do not need.

At the moment I don't have full testcase which can defend all this method.

`putWorker()` is not needed, because export works OK, and other PUTs not need any lazy NL.

`putField()` should have `lazyTrailingNL`.

For others I'm not sure atm.

3. Please describe the testing you have done to check if `DISPLAY`, `VIEW`, `MESSAGE` and similar redirected terminal usage need the `lazyTrailingNL` support.

covered by testcases in `testcases/uast/no_echo`.

4. Please check in your testcases for the PUT and EXPORT `lazyTrailingNL` testing.

Export added NL always. either if it is put next, covered by testcase.

5. Please look at how `writeNewLine()` is used in `endOutput()` and `processLineEnd()`. When you use it, don't you also need to call `rawIncrementNextLineNum()`?

yes, I've added `rawIncrementNextLineNum()`; too.

6. In `StreamFactory` and `StreamDaemon`, please merge the 2 history entries (one by me and one by you) into a single history entry. You can put your initials there.

done

7. Did you test the `uast/testcases/redirected_*.p` tests as I requested in note 100? What are the results of that testing?

yes, I have results:

`redirected_editor*.p`
`redirected_dynamic_down_frame.p`
`redirected_frame_conditional_down.p`
`redirected_frame_footer_main.p`
`redirected_output_training1.p`
`redirected_output_training0.p`

Tests worked properly.

`redirected_boxed_frame.p` found a bug in `RedirectedTerminal`. For last line in output idx calculated not properly. Now it is fixed.

`redirected_report*`:
`redirected_report6_3.p` is not converted properly
`redirected_report6_4.p` is not converted properly
`redirected_report6_5.p` is not converted properly
`redirected_report6_6.p` is not converted properly
`redirected_report6_7.p` is not converted properly
So I don't have results. Converted code for these tests are not compiled:

```
FrameElement[] elementList1 = new FrameElement[]
{
    new WidgetElement(f1Frame.widgetj()),
    new WidgetElement() // no empty constructor
};
```

#123 - 01/15/2015 05:21 AM - Evgeny Kiselev

At the moment regression tests finished with failed status.

5 tests are failed because of different numbers of rows (always difference is 1 in one tests is 2, e.g. 186 rows, but need 185).

So some case in this tests are not cover in my code.

I want to specified again algorithm of lazy NL.

- NL should be added after any PUT (formatted/unformatted) with open unnamed output stream and if next output is message.

or

- Message should be always on New Line if current position is not in the start of line. I think it's more correct. But when we work with Window class or with RedirectedTerminal we have lost all information about data (we don't know if it was message or any other data). And then I try to use `com.goldencode.p2j.util.Stream#writeControlled` I can add some NL symbol with different type of data (not message).

Conclusion:

- We need know about Stream position in Window or RedirectedTerminal to add needed NL symbol and handle case, that message should start from start of line(I think it's not very good).

- Stream should know about data, we need special flag or special method which should handle message case.

#124 - 01/15/2015 09:38 AM - Greg Shah

I think your idea of changing the approach seems to be on the right track.

Message should be always on New Line if current position is not in the start of line. I think it's more correct.

I agree, this seems to be a better way to think about the problem.

I've asked Constantin to review this and provide some thoughts.

#125 - 01/16/2015 02:11 PM - Constantin Asofiei

Evgeny Kiselev wrote:

- Message should be always on New Line if current position is not in the start of line. I think it's more correct. But when we work with Window class or with RedirectedTerminal we have lost all information about data (we don't know if it was message or any other data). And then I try to use `com.goldencode.p2j.util.Stream#writeControlled` I can add some NL symbol with different type of data (not message).

I think this is the correct approach. MESSAGE should output a new line as needed. Have you determined what happens if the current line contains only white space and the stream cursor is not on the first column? Does the MESSAGE output bellow this white-space line or the cursor is just repositioned on the first column? Use a boxed frame to test (as a NL it's always output after it, as I recall) or a PUT statement ending with `SKIP(1) SPACE(<number of spaces>)`.

- We need know about Stream position in Window or RedirectedTerminal to add needed NL symbol and handle case, that message should start from start of line(I think it's not very good).

Why do you think is not very good? I agree that Stream might need some changes so that it knows when the cursor is on a new line; also, this will isolate this problem to the one who needs, the MESSAGE stmt in this case.

Please merge your update with the latest b2r and post it here.

#126 - 01/17/2015 05:57 PM - Evgeny Kiselev

Constantin Asotiei wrote:

Evgeny Kiselev wrote:

- Message should be always on New Line if current position is not in the start of line. I think it's more correct. But when we work with Window class or with RedirectedTerminal we have lost all information about data (we don't know if it was message or any other data). And then I try to use com.goldencode.p2j.util.Stream#writeControlled I can add some NL symbol with different type of data (not message).

I think this is the correct approach. MESSAGE should output a new line as needed. Have you determined what happens if the current line contains only white space and the stream cursor is not on the first column? Does the MESSAGE output below this white-space line or the cursor is just repositioned on the first column? Use a boxed frame to test (as a NL it's always output after it, as I recall) or a PUT statement ending with SKIP(1) SPACE(<number of spaces>).

I was a little bit wrong, see next testcase:

```
output to value ("message.txt").
put unformatted "unformat~n".
message "message:test1".
put unformatted " ".
message "message:test2".
put unformatted SKIP(1) SPACE(10).
message "message:test3".
put unformatted "this is the appended text".
output close.
```

P2J output:

```
unformat
NL
message:test1
' '
message:test2
NL
' '
message:test3
this is the appended text
```

I've marked NL as a new line and ' ' as a whitespace.

So we need always NL before message if message is not on the first line(or we need to calculate NL character as not 0 column). Also it's not clear for me, why it was double NL after message:test2 ?

Updated:

We calculate column in Stream properly.

```
put unformatted "unformat~n".
```

NL at the end will set column to 1.

#127 - 01/18/2015 09:11 AM - Greg Shah

Please post the corresponding 4GL output so that I can better understand what this code is supposed to do.

#128 - 01/18/2015 09:06 PM - Evgeny Kiselev

Greg Shah wrote:

Please post the corresponding 4GL output so that I can better understand what this code is supposed to do.

Sorry I did mistake, it was 4GL output:

```
unformat
NL
message:test1
' '
message:test2
NL
' '
message:test3
this is the appended text
```

Here is P2J output from latest revision.(Note that in latest revision there is no message:test1)

```
unformat ' 'message:test2
NL
' 'message:test3
this is the appended text
```

#129 - 01/19/2015 01:27 PM - Greg Shah

So we need always NL before message if message is not on the first line(or we need to calculate NL character as not 0 column).

I think instead of "first line" you mean "first column".

Please look closely at `ThinClient.sendToStream()`:

```
public int sendToStream(String message)
{
    int redir;
    Stream out = streamFromId(currentStream);

    // if the terminal is redirected, then send this message to the
    // stream
    if (out != null && out.isUnnamed())
    {
        // assumes that we are in a reset state
        ((ChuiOutputManager) tk).append(message, Color.DEFAULT);
        flushStream(out);
    }

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

    // switch back to interactive mode temporarily, to allow the message
    // to be displayed on the terminal
    redir = switchOutput(-1, false);
    return redir;
}
```

In this case, the output is redirected and the destination is unnamed. In other words you used output to `value()`, which is an unnamed stream instead of output stream `my-stream` to `value()`, which is a "named" stream (in this case named "my-stream").

That means that the code executes this:

```
{
    // assumes that we are in a reset state
    ((ChuiOutputManager) tk).append(message, Color.DEFAULT);
    flushStream(out);
}
```

Note this "assumes that we are in a reset state". I guess this explains why message is behaving badly in this case.

Is the answer to this problem, just to give the stream a chance to flush BEFORE the `append()`? If there is nothing buffered, then it should do nothing. In other words, if it is already "reset", then it would not flush but instead just return.

`ThinClient.sendToStream()` is only used for MESSAGE, so any such change seems safe.

Also it's not clear for me, why it was double NL after `message:test2` ?

SKIP (with no expression) is a kind of conditional newline. If there is text on the line before the SKIP, then there is a newline added, otherwise it is ignored.

It recall that `SKIP(1)` is an unconditional newline, but before it puts in the newline it will implicitly execute the SKIP. The use of SKIP does the same thing in interactive frame layout too. In other words, `SKIP(1)` outputs two newlines in this case, even though it is counter-intuitive.

#130 - 01/19/2015 09:18 PM - Evgeny Kiselev

- File 01_evk_upd20150119a.zip added

Explicit flush stream is helped, thanks.

That's why different test in regression testing were failed from time to time. I've cleaned lazy NL from Stream class, because it's not necessary. This update has one TODO in RedirectedTerminal, I will delete it later.

Regression testing is running now, but I think it will passed this time.

#131 - 01/20/2015 07:26 AM - Greg Shah

Code Review evk_upd20150119a.zip

It looks good.

Are the ChUI-specific changes to Window.message() still needed (to set the cursor position to column 0)? If so, please put in a TODO that the ChUI specific code should be moved into the ChUI subclass.

#132 - 01/23/2015 07:46 AM - Evgeny Kiselev

- File 01_evk_upd20150121a.zip added

Greg Shah wrote:

Are the ChUI-specific changes to Window.message() still needed (to set the cursor position to column 0)? If so, please put in a TODO that the ChUI specific code should be moved into the ChUI subclass.

Changes in Window class is still needed. But changes in RedirectedTerminal is not needed now. Looks like boxing calculated properly now.

At the moment, there are 3 tests failed:

1. gso_276 - incorrect calculation of pages:

```
Record(s) selected: 167 ..... 25 Pages (but should be 24)
```

2. gso_17 - one extra line with data in gso_17.sum. Don't have any idea why.

3. gso_488 - different data.
should be:

```
Active/Inactive,Item,Description,Net Qty Used,U/M,Cov U/M,Unit Weight,Weight Used,Hazmat Code  
A,0A0004CJE,LIQUID NITROGEN - BULK TANK -COMPOSITE,2391927,CF,,0,0,FHPR,
```

But found('yes' instead of 'A'):

```
Active/Inactive,Item,Description,Net Qty Used,U/M,Cov U/M,Unit Weight,Weight Used,Hazmat Code  
yes,0A0004CJE,LIQUID NITROGEN - BULK TANK -COMPOSITE,2391927,CF,,0,0,FHPR,
```

Does reference results have been taken from 4GL output ?

#133 - 01/23/2015 08:39 AM - Greg Shah

Code Review evk_upd20150121a.zip

The changes are fine. In the future, please post the update without the 01_ prefix.

About the test failures:

For gso_276 and gso_17, you will have to debug in to see the causes.

But I can guess the cause of the gso_488 was caused by the removal of format string processing from the Stream.generateFormattedText().

Does reference results have been taken from 4GL output ?

If this is a report, then it definitely is taken from the 4GL. Almost all screens are taken from the 4GL too, but there are some exceptions.

Eugenie Lyzenko: is there anything about GSO 488 which was not captured in the 4GL?

You can always cut down the line of code that is generating the report detail and create a simple 4GL testcase from it. Then test that on lindev01.

#134 - 01/23/2015 09:29 AM - Eugenie Lyzenko

Eugenie Lyzenko: is there anything about GSO 488 which was not captured in the 4GL?

I think yes. The 4GL system dragonfly access was stopped at 04/23/2010, right? The GSO 488 was introduced at 08/30/2010. This is later. However we certainly know how the 4GL reports should be for this test(especially *.csv file).

#135 - 01/25/2015 09:01 PM - Evgeny Kiselev

- File evk_upd20150126a.zip added

Tests gso_17 and gso_488 has been fixed in Stream class. Even if output is unformatted we still need to use formatting if it was specified explicit.

Updated is merged with latest revision.

#136 - 01/26/2015 07:46 AM - Evgeny Kiselev

My changes in Window class is issue for gso_276 test. Looking for approach.

#137 - 01/26/2015 12:34 PM - Greg Shah

Code Review evk_upd20150126a.zip

1. Please remove import org.apache.commons.logging.*; from Stream.java.
2. The change to Stream.generateFormattedText() does not seem quite right.
 - The approach you are taking only will work for character types. Other format strings will never match. For example, decimal default format of ->>, >>9.99 or logical default format of yes/no.
 - Use BaseDataType.defaultFormatString() to obtain the real, system-specific default format string for a type. For example, the format for decimal will have the locale-specific group and decimal separator characters.
 - Even the character type seems incorrect as the default format for character is x(8). It is not based on the length of the contained text.

#138 - 01/26/2015 12:35 PM - Greg Shah

Evgeny Kiselev wrote:

My changes in Window class is issue for gso_276 test. Looking for approach.

Did you intend to post a version of Window? Nothing was included.

#139 - 02/05/2015 08:47 PM - Evgeny Kiselev

- File gso_276.txt added
- File gso_276_2.txt added
- File evk_upd20150205a.zip added

Even the character type seems incorrect as the default format for character is x(8). It is not based on the length of the contained text.

Yes, default format is x(8) but constructor for com.goldencode.p2j.util.PutField#PutField(java.lang.String) specified format explicit. I've added specific case for character type.

I've added two output files for test gso_276. Content in this files is the same. But in some lines before or after page headers. This issue depends on ChuiOutputManager.cursorInvalid and RedirectedTerminal.reset variables.
If I'm rollback changes in Window class (this changes actually do nothing at the moment, because for RT class cursor position is always 0:0) But this code changes state of cursorInvalid in ChuiOutputManager and for example I'll always lost first message because of code in ChuiOutputManager.append().
I've tried different approaches but nothing helped. I'm always losing NL or received wrong output.
Other tests are passed.

#140 - 02/11/2015 09:39 PM - Greg Shah

Code Review evk_upd20150205a.zip

The changes look fine. You'll need to merge up to the latest level before the next set of tests.

This issue depends on `ChuiOutputManager.cursorInvalid` and `RedirectedTerminal.reset` variables. If I'm rollback changes in `Window` class (this changes actually do nothing at the moment, because for `RT` class cursor position is always 0:0) But this code changes state of `cursorInvalid` in `ChuiOutputManager` and for example I'll always lost first message because of code in `ChuiOutputManager.append()`. I've tried different approaches but nothing helped. I'm always losing `NL` or received wrong output.

I don't know the answer here. But I am thinking about the following approach:

1. Which changes are causing this? I suspect that only the `Window`, `ThinClient` and `Stream` can have anything to do with this. Even the `Stream` change is now probably not the cause because it really just should change whether some output is formatted or not AND it really should not be visible in any the `MAJIC` tests or we would have seen the need for it already. On `devsrv01`, create a fresh `P2J` check out and apply only the `ThinClient` and `Window` changes. Then bring up a server and manually run the `GSO 276` test (using the `gso_276.xml`, manually enter the inputs as a user). Compare the generated report and confirm that the problem exists with only those changes involved. Then remove the `Window` change, rebuild and run the test again. Can the problem be isolated to only the `ThinClient` change?

2. Once you isolate the change that is the cause, you can try a debugging session. Download the `MAJIC` sources, jars and `run/` directories to your local system. Set the database URL to point to your tunnel for your `devsrv01` database instance. Using this, you can debug `MAJIC`. You now can set a conditional breakpoint that triggers when you get to the failing record. Step through the output code on the client side and watch the state to try to figure out where things go wrong.

Constantin: do you have any better ideas?

#141 - 03/27/2015 06:51 PM - Evgeny Kiselev

I think I'm tried to fix wrong bug. The root of my problem is some other bug in latest revision code. At the moment I'm focus on this problem. Here is testcase:

```
output to value ("message.txt").
put unformatted "unformat~n".
message "message:test1".
put unformatted " ".
message "message:test2".
put unformatted SKIP(1) SPACE(10).
message "message:test3".
put unformatted "this is the appended text".
output close.
```

```
output to value ("display.txt").
put unformatted "unformat~n".
display "message:test".
```

```
put unformatted "this is the appended text".
output close.
```

```
output to value ("export.txt").
put unformatted "unformat~n".
export "message:test".
put unformatted "this is the appended text".
output close.
```

4gl output for message.txt file:

```
unformat
message:test1
message:test2
message:test3
this is the appended text
```

4gl output in latest revision:

```
unformat  message:test2
          message:test3
this is the appended text
```

My fixes fixed the problem with formatting and text placing but hide the root of problem that message:test1 was just disappeared. Debugging of regression server didn't show me the problem. The problem could be in Stream, RedirectedTerminal, or ThinClient classes. I think if I fix problem in latest release it will help to fix problem in my code.

#142 - 03/30/2015 09:01 PM - Evgeny Kiselev

- File 01_evk_upd20150330a.zip added

fix issue described in note 141.

if variable `cursorInvalid` from `ChuiOutputManager` is true with border cursor values, than some messages have been skipped in `ChuiOutputManager.append(java.lang.String, com.goldencode.p2j.ui.Color)`
regression testing is not finished at the moment.

#143 - 03/31/2015 10:47 AM - Greg Shah

Code Review `evk_upd20150330a.zip`

This seems risky. There are a large number of possible changes this will cause, depending on how many callers depend on that code to return false. My concern is that I can't predict all the ways we use that code and I suspect that we don't want the cursor to be considered valid when it is past the end of the screen.

I've asked Constantin to comment as well.

Also: please only post and/or distribute updates as `evk_upd20150330a.zip` instead of `01_evk_upd20150330a.zip`.

#144 - 03/31/2015 02:11 PM - Constantin Asofiei

Evgeny, I think a better test is using just a MESSAGE stmt and an unnamed stream, like this:

```
output to value ("message.txt").
message "message:test1".
output close.
```

`message.txt` will be blank in P2J. And the root cause looks like is that after the message is displayed on screen, the cursor is invalid as we draw the entire line, 80-char width length (thus the last char drawn moves the cursor to column 80, which makes the `cursorInvalid` flag true). When we try to send the same text to the stream, the cursor remains invalid, thus `append` fails.

What I think we need is to move the `cursorInvalid` field from the `ChuiOutputManager` class to the `BasePrimitives` class, as this is set only after we call `BasePrimitives.cursorAt` or `BasePrimitives.cursorStay` - thus, when the output is switched, the flag state is inherited from the previous terminal, which is incorrect. We need to allow the `cursorInvalid` flag to be tracked with the output destination in use, and not just use a global flag.

#145 - 05/05/2015 09:33 AM - Evgeny Kiselev

- File `evk_upd20150504a.zip` added

merged with latest revision

moved `cursorInvalid` to the `BasePrimitives`

But I couldn't test it now (after merged 29.04.2015 `hc_upd20150428b.zip`) I got null pointer in testcase from note 141

```
Caused by: java.lang.NullPointerException
    at com.goldencode.p2j.ui.chui.FillInImpl.nativeWidth(FillInImpl.java:314)
    at com.goldencode.p2j.ui.client.FillIn.updateSize(FillIn.java:2559)
    at com.goldencode.p2j.ui.client.FillIn.initialize(FillIn.java:584)
```

```
at com.goldencode.p2j.ui.client.FillIn.initialize(FillIn.java:414)
at com.goldencode.p2j.ui.client.WidgetRegistry.reconstructWidget(WidgetRegistry.java:120)
at com.goldencode.p2j.ui.client.WidgetRegistry.pushDefinition(WidgetRegistry.java:364)
at com.goldencode.p2j.ui.chui.ThinClient$18.run(ThinClient.java:7126)
at com.goldencode.p2j.ui.chui.ThinClient.eventBracket(ThinClient.java:13116)
at com.goldencode.p2j.ui.chui.ThinClient.eventDrawingBracket(ThinClient.java:13065)
at com.goldencode.p2j.ui.chui.ThinClient.pushOneDef(ThinClient.java:7114)
at com.goldencode.p2j.ui.chui.ThinClient.pushScreenDefinition(ThinClient.java:7089)
at sun.reflect.GeneratedMethodAccessor9.invoke(Unknown Source)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:606)
```

I've read all emails but didn't find any special things that I should do. I do not have any changes in widgets.

#146 - 05/06/2015 02:26 AM - Constantin Asofiei

Evgeny Kiselev wrote:

merged with latest revision
moved cursorInvalid to the BasePrimitives
But I couldn't test it now (after merged 29.04.2015 hc_upd20150428b.zip) I got null pointer in testcase from note 141
[...]
I've read all emails but didn't find any special things that I should do. I do not have any changes in widgets.

I've tested with latest rev (10847) and the note 141 test works fine for me (no abends). All files are OK except display.txt which is missing an empty line.

With AspectJ being introduced, ensure that you are either using the proper p2j.jar or your IDE has AspectJ integrated.

#147 - 05/06/2015 12:16 PM - Evgeny Kiselev

Thx, I've fixed it. now it's working fine for if I worked from p2j.jar instead of using code.

#148 - 05/06/2015 02:26 PM - Greg Shah

Code Review evk_upd20150504a.zip

1. import org.apache.commons.logging.*; should not be added to Stream.java.
2. Please merge with bsr 10848.

At this point only GSO 276 still fails? All other issues are resolved?

#149 - 05/07/2015 03:36 PM - Evgeny Kiselev

- File evk_upd20150506a.zip added

All other issues are resolved. All tests has been passed if I exclude changes from Window and ThinClient classes (except case from note 141). GSO 276 is still fails. I have experimental version which is work properly, but it's dirty version and I'm still testing it.

I'm using next idea:

Changes in Window and ThinClient classes are not right because NL we need only in case then we have some put unformatted and display/message output. But put unformatted handled only in Stream class, and Window and ThinClient know nothing about this operation. Which why I think this problem should be handled in Stream class. Idea that Stream.writeControlled should always printed on NL if current position is not NL. trailingNL is not applicable to do this, because it handles NL from put unformatted cases, but it's not correct.

#150 - 05/08/2015 11:11 AM - Greg Shah

Good, this sounds promising. Please do implement the changes so that they are nicely separated (e.g. no ChUI/GUI-specific stuff in common UI code). Post the result. Hopefully it can get through regression testing and checked in.

#151 - 07/13/2015 01:55 PM - Evgeny Kiselev

- File evk_upd20150713a.zip added

All changes except Stream class is passed regression testing.

Changes in Stream class is not fully correct. gso_103 is not worked correctly.

I couldn't find a good solution for all cases. If I fix one problem, next problem is occure.

Main idea is described in note 149. I couldn't find good solution to trail NL.

I'm sure, that this problem should be fixed in Stream class or in another class, but it should be on server side. Client side couldn't track put output.

Any other changes (except Stream class) are ready to commit.

#152 - 03/25/2016 12:41 PM - Constantin Asofiei

Created task branch 1628a from trunk rev 10985.

Rev 10986 contains the merged evk_upd20150713a.zip (there are still some javadocs missing, will fix in the final review/fix).

Runtime testing will start.

#153 - 03/28/2016 07:45 AM - Constantin Asofiei

Rebased trunk rev from 1628a rev 10986.

1628a rev 10988 fixes a PUT issues (do not emit NL before PUT if cursor is on column 0).

Runtime testing started again.

#154 - 03/28/2016 08:47 AM - Greg Shah

Both the merge of the evk_upd20150713a.zip into 1628a and the changes in 10988 are good.

#155 - 03/29/2016 05:23 AM - Constantin Asofiei

1628a rev 10989 fixes two more issues (format for unknown date variables and a NL emitted before the page headers).

#156 - 03/29/2016 01:16 PM - Constantin Asofiei

Branch 1628a passed runtime testing and was merged to trunk rev 10987 and archived.

#157 - 03/29/2016 01:20 PM - Greg Shah

- % Done changed from 0 to 100
- Assignee changed from Evgeny Kiselev to Constantin Asofiei
- Status changed from New to Closed

#158 - 11/16/2016 12:06 PM - Greg Shah

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

#159 - 11/22/2016 10:03 AM - Greg Shah

- File deleted (gso_276.txt)

#160 - 11/22/2016 10:03 AM - Greg Shah

- File deleted (gso_276_2.txt)

Files

evk_upd20140502a.zip	180 KB	05/02/2014	Evgeny Kiselev
evk_upd20140509a.zip	180 KB	05/09/2014	Evgeny Kiselev
evk_upd20140607a.zip	200 KB	06/09/2014	Evgeny Kiselev
ges_upd20140612a.zip	300 KB	06/12/2014	Greg Shah
evk_upd20140616a.zip	466 KB	06/17/2014	Evgeny Kiselev
evk_upd20140624a.zip	466 KB	06/24/2014	Evgeny Kiselev
failed1.png	71.5 KB	07/03/2014	Evgeny Kiselev
faield2.png	86.4 KB	07/03/2014	Evgeny Kiselev
evk_upd20140717a.zip	484 KB	07/17/2014	Evgeny Kiselev
01_evk_upd20140721a.zip	504 KB	07/21/2014	Evgeny Kiselev
01_evk_upd20140721b.zip	504 KB	07/21/2014	Evgeny Kiselev
last_test1.png	9.33 KB	07/24/2014	Evgeny Kiselev
last_test2.png	14.1 KB	07/24/2014	Evgeny Kiselev
evk_upd20140728a.zip	509 KB	07/29/2014	Evgeny Kiselev
evk_upd20140731a.zip	541 KB	07/31/2014	Evgeny Kiselev
evk_upd20140801a.zip	541 KB	08/01/2014	Evgeny Kiselev
evk_upd20140803a.zip	541 KB	08/04/2014	Evgeny Kiselev
01_evk_upd20140805a.zip	541 KB	08/08/2014	Evgeny Kiselev
01_evk_upd20140812a.zip	305 KB	08/15/2014	Evgeny Kiselev
01_evk_upd20140820a.zip	429 KB	08/26/2014	Evgeny Kiselev
01_evk_upd20140910a.zip	478 KB	09/10/2014	Evgeny Kiselev
ca_upd20140917a.zip	479 KB	09/17/2014	Constantin Asofiei
01_evk_upd20141003a.zip	476 KB	10/07/2014	Evgeny Kiselev
evk_upd20141216a.zip	372 KB	12/18/2014	Evgeny Kiselev
01_evk_upd20150119a.zip	500 KB	01/20/2015	Evgeny Kiselev
01_evk_upd20150121a.zip	492 KB	01/23/2015	Evgeny Kiselev

evk_upd20150126a.zip	492 KB	01/26/2015	Evgeny Kiselev
evk_upd20150205a.zip	493 KB	02/06/2015	Evgeny Kiselev
01_evk_upd20150330a.zip	3.7 KB	03/31/2015	Evgeny Kiselev
evk_upd20150504a.zip	511 KB	05/05/2015	Evgeny Kiselev
evk_upd20150506a.zip	511 KB	05/07/2015	Evgeny Kiselev
evk_upd20150713a.zip	332 KB	07/13/2015	Evgeny Kiselev