# User Interface - Bug #3691

## literal $ character in format appears on the right side of the fill-in

08/21/2018 10:51 AM - Greg Shah

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

**Description**

**Related issues:**

| | |
|---|---|
| Related to Base Language - Bug #4123: conversion generates incorrect java cod... | **Closed** |
| Related to User Interface - Bug #4130: FWD truncates decimal part instead of ... | **New** |

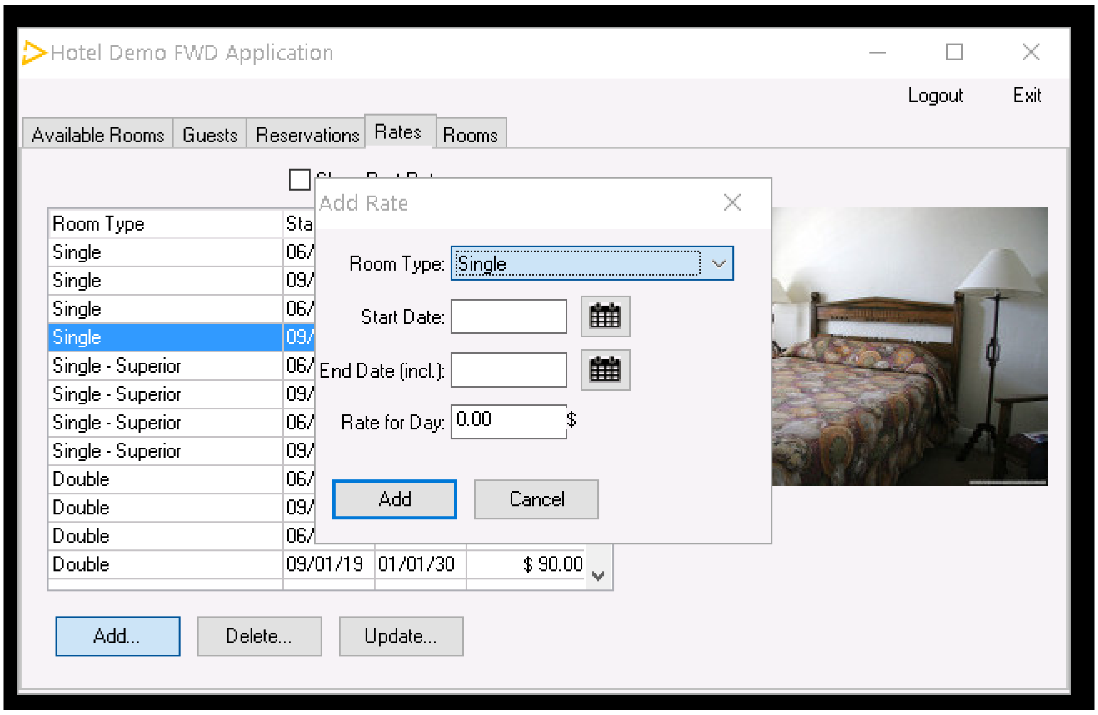
### History

**#1 - 08/21/2018 10:57 AM - Greg Shah**

*- File hotel_gui_trunk_rev_11281_format_string_literal_in_wrong_place_20180821.png added*

Recreate in Hotel GUI:

1. Login
2. Click on the Rates tab.
3. Select any room type in the browse.
4. Click Add button.
5. The "$" part of the format is displayed (incorrectly) on the right side of the "Rate for Day" widget.



| Room Type | Sta | | |
|---|---|---|---|
| Single | 06/ | | |
| Single | 09/ | | |
| Single | 06/ | | |
| Single | 09/ | | |
| Single - Superior | 06/ | | |
| Single - Superior | 09/ | | |
| Single - Superior | 06/ | | |
| Single - Superior | 09/ | | |
| Double | 06/ | | |
| Double | 09/ | | |
| Double | 06/ | | |
| Double | 09/01/19 | 01/01/30 | $ 90.00 |

Room Type: Single
Start Date:
End Date (incl.):
Rate for Day: 0.00 $

This bug occurs in all GUI clients (Swing, web virtual desktop and web embedded).

**#2 - 04/30/2019 05:11 PM - Greg Shah**

*- Assignee set to Alexei Kaigorodov*

**#3 - 05/08/2019 09:00 AM - Greg Shah**

From Alexei:

> Installed the Hotel demo and made sure the bug exists, Now thinking how to find the place where the error string is constructed.

For widgets, the display formatting (and format strings) are handled by om/goldencode/p2j/ui/client/format/DisplayFormat and its child classes.

For non-display purposes, handling format strings is done in the BaseDataType.toString() implementations, with each type having some unique behavior.

In all format strings there are characters that describe the format of specific positions in the output string. For example, a >>>9.99 decimal format string will cause there to always be output in the 2 places to the right of the decimal point and in the 1 digit to the immediate left of the decimal point. It will allow output up to 4 digits on the right side of the decimal point, but these are optional. If the value doesn't need them, they will not appear. You can also have "fixed" characters that are just text and are included in the output without any transformation based on the value of the variable. A decimal format $>>>9.99 would be an example. In this case, the $ dollar sign is just text and should always appear on the left side of the output, no matter the variable or widget's value.

I recommend finding the original format string for the widget in question. Then you can look at how the referenced classes above handle it.

**#4 - 05/09/2019 08:55 AM - Alexei Kaigorodov**

I saw in debugger that the format string is not changed (and does not contain any dollar sign).

There can be 2 possible reasons: the string with rate value is appended with '$' and it goes beyond the border, or the border itself is corrupted. To determine. I need to find where the actual image is constructed.

**#5 - 05/09/2019 09:06 AM - Greg Shah**

> the format string is not changed (and does not contain any dollar sign)

What is the 4GL code that has been converted for this widget? What is the original format string that was specified?

> I need to find where the actual image is constructed.

By "actual image", I guess you mean "widget drawing". Each widget type will have a draw() method that handles this. It can sometimes be in a parent class. The widget type in this case should be FillInGuiImpl.

**#6 - 05/09/2019 09:14 AM - Alexei Kaigorodov**

the original code is in file update-rate-dialog.w:

```
DEFINE VARIABLE roomRate AS DECIMAL FORMAT ">>>,>>9.99":U INITIAL 0
LABEL "Rate for Day"
VIEW-AS FILL-IN
SIZE 14 BY 1 NO-UNDO.
```

the converted code is in file src/com/goldencode/hotel/ui/UpdateRateDialogGdialog.java:

```
FillInWidget roomRate = new FillInWidget();
...
roomRate.setFormat(">>>,>>9.99");
roomRate.setLabel("Rate for Day");
roomRate.setWidthChars(14);
roomRate.setHeightChars(1);
```

**#7 - 05/09/2019 10:45 AM - Greg Shah**

Is there any use of the $ character in that program (check the .cache file which has been fully preprocessed)?

**#8 - 05/09/2019 11:29 AM - Alexei Kaigorodov**

There any many use of the $ character .cache files, but not in update-rate-dialog.w.cache (no changes there compared to update-rate-dialog.w), and not with '$" in the end of format string.

the full list of occurrences:

```
guests-frame.w.cache
stay.price COLUMN-LABEL "Room Price" FORMAT "$ >>>,>>>,>>9":U
"stay.price" "Room Price" "$ >>>,>>>,>>9" "integer" ? ? ? ? ? ? no ? no no ? yes no no "U" "" "" "" "" "" "" 0 no 0 no no
rates-frame.w.cache
rate.rate FORMAT "$ >>>,>>9.99":U
"rate.rate" ? "$ >>>,>>9.99" "decimal" ? ? ? ? ? ? no ? no no ? yes no no "U" "" "" "" "" "" "" 0 no 0 no no
STRING + " ($" +
reservations-frame.w.cache
reservation.price COLUMN-LABEL "Room Price" FORMAT "$ >>>,>>9.99":U
"reservation.price" "Room Price" "$ >>>,>>9.99" "decimal" ? ? ? ? ? ? no ? no no "15.8" yes no no "U" "" "" "" "" "" "" 0 no 0 no no
update-rate-dialog.w.cache
"$" VIEW-AS TEXT
" ($" + STRING + ")!"
update-reservation-dialog.w.cache
txt = trim(string(price, "$ >>>,>>9.99")).
update-service-dialog.w.cache
"$" VIEW-AS TEXT
update-stay-dialog.w.cache
DEFINE VARIABLE totalPrice AS INTEGER FORMAT "$ ->,>>>,>>9.99":U INITIAL 0
DEFINE VARIABLE totalServices AS INTEGER FORMAT "$ ->,>>>,>>9.99":U INITIAL 0
TRIM) + ")?"
txt = trim(string(price, "$ >>>,>>9.99")).
abl/common/adm2/smart.p.cache
IF pcDelim = cUnique THEN cUnique = "$":U.      /* use alternative char*/
```

**#9 - 05/09/2019 11:45 AM - Greg Shah**

Focus on this one: "$" VIEW-AS TEXT. Perhaps the problem is that it is being placed in the wrong location?

**#10 - 05/09/2019 11:52 AM - Constantin Asofiei**

Alexei, if you compile FWD with a -Dwidget.browser=true at the ant command, then you will have a widget browser with the entire UI tree. A swing window will be opened for every legacy window opened by FWD - use the left-side tree to find the widget associated with the $ text in the window - you will see it highlighted in a red overlay. More details are in #2412 about the widget browser.

You will need to use the FWD Swing GUI client for this - the Web client will not be able to show you the widget browser.

**#11 - 05/09/2019 10:54 PM - Alexei Kaigorodov**

Constantin, https://proj.goldencode.com/projects/internal-documentation/wiki/Getting_Started_with_FWD_Development
says build must be done by ./gradlew core, not ant.
However, I ran

```
ant -Dwidget.browser=true core
cd deploy/server
server.sh
```

and opened https://localhost:41379/index.html, but no swing window opened.

**#12 - 05/10/2019 01:00 AM - Eric Faulhaber**

Alexei Kaigorodov wrote:

> Constantin, https://proj.goldencode.com/projects/internal-documentation/wiki/Getting_Started_with_FWD_Development
> says build must be done by ./gradlew core, not ant.
> However, I ran
> [...]
> and opened https://localhost:41379/index.html, but no swing window opened.

I'm not sure where that URL came from. To run the widget browser, you must run the Swing client, which is not loaded from a browser. Temporarily modify hotel_gui/deploy/client/client.sh and add the system property -Dwidget.browser=true somewhere in the java command before the class name. Then:

```
cd <location of p2j project>
./gradlew -Dwidget.browser=true core
cd <location of hotel_gui project>/deploy/server
./server.sh
```

In a separate terminal...

```
cd <location of hotel_gui project>/deploy/client
./client.sh
```

Hynek, Constantin, this is the first time I've run the widget browser, so modifying the client.sh was a bit of a hack. I guess you guys launch the Swing client from your IDEs, so you don't use client.sh? Anyway, I think that script needs to be modified to make the widget browser a simple option.

**#13 - 05/10/2019 05:57 AM - Hynek Cihlar**

Eric Faulhaber wrote:

> Hynek, Constantin, this is the first time I've run the widget browser, so modifying the client.sh was a bit of a hack. I guess you guys launch the Swing client from your IDEs, so you don't use client.sh? Anyway, I think that script needs to be modified to make the widget browser a simple option.

I either add the widget.browser Java property to the launch configuration in my IDE or modify client.sh. New parameter for client.sh would be handy indeed.

Btw. you can run Widget Browser for the web client, too. Obviously you have to run FWD client on localhost. Add -Dwidget.browser to jvmArgs directory value in clientConfig container, remove java.awt.headless from the same directory value and comment out jvm = jvm + pad + "-Djava.awt.headless=true" in ClientBuilderOptions.

**#14 - 05/10/2019 07:05 AM - Alexei Kaigorodov**

Greg, you are right. The source file update-rate-dialog.w has lines:

```
DEFINE FRAME gDialog
...
     "$" VIEW-AS TEXT
          SIZE 8 BY .62 AT ROW 6.1 COL 31 WIDGET-ID 10
```

which draw String "$       " on the dialog window, which is seen in the widget browser.
I propose to remove it entirely, and insert '$' into the label of the rate field:

```
     LABEL "Rate for Day ($)"
```

**#15 - 05/10/2019 07:37 AM - Greg Shah**

This does suggest there **may not** be a FWD bug here.

Before we consider any 4GL code changes, I want to **make sure** there is no FWD bug here to fix.  I think we need to see the code running in the 4GL to compare it.  But it is quite a bit of effort to get the Hotel GUI code fully running in the 4GL, because you have to build a database and get the configuration right to make it work.  As an alternative, please extract the frame definition (and the code that displays it) and simply use that by itself to see the result.  You don't need to use any data because we are just checking on the layout.

If there is no bug in FWD, then we can modify the 4GL.  Instead of changing the label, I prefer to implement it in the format string ("$ >>>,>>9.99" instead of ">>>,>>9.99").  This is the more normal way to do this and it might be easier to internationalize as well.

If there is a FWD bug, we can discuss it here.

**#16 - 05/13/2019 02:06 AM - Alexei Kaigorodov**

*- File bug3691_Dollar_Sign.p added*

*- File UpdateRateTest.jpg added*

I created a test program which runs on OpenEdge. Its visualization is exactly the same as on FWD. See the attached files.

**#17 - 05/13/2019 04:57 AM - Alexei Kaigorodov**

*- File Fixed_4GL.png added*

*- File Fixed_FWD.png added*

So the correction is to remove drawing of the "$" sign as separate string and replacing the format string ">>>,>>9.99" with of "$ >>>,>>9.99". However, this reveals a small bug in FWD: the dollar sign is not at the leftmost position, as it is when running on OpenEdge (see screenshots).

Do I need to investigate the reason of this incompatibility?

**#18 - 05/13/2019 08:11 AM - Greg Shah**

> Do I need to investigate the reason of this incompatibility?

Yes, please do.

**#19 - 05/14/2019 05:26 AM - Alexei Kaigorodov**

I made a fix to hotel_gui/abl/update-rate-dialog.w and commited it (rev. 198).

**GES: The following notes are moved from #3631.  They were incorrectly added there.  I am inserting them into this note to maintain the proper order of the discussion.**

Alexei wrote on May 24, 2019:

> tried to find out the place where format "$ >>>,>>9.99" is used to show value "  $ 0.00  " but failed. The widget is defined in file hotel_gui/src/com/goldencode/hotel/ui/UpdateRateDialogGdialog.java:

```
68:     FillInWidget roomRate = new FillInWidget();
```

but attempts to discover the place where the format is retrieved were not successfull. I could not even determine if the string is formatted on server side or client side.

> Any clues?

Hynek responded on May 24, 2019:

> Check FillIn.getScreenValue() and FillIn.getValue(). Put a break point in these methods to see when are they called.

Greg responded on May 24, 2019:

> Also, the formatting of the text for the screen would be done in com/goldencode/p2j/ui/client/format/NumberFormat.java.

**#20 - 05/27/2019 10:54 AM - Alexei Kaigorodov**

Narrowed the suspicious number formatting to the line in in the method FillIn#display():

962:  config.appDataPres = config.appFormat.fromVar(appVar);

here config.appFormat.formatDef = "$ >>>,>>9.99"
appVar.toString() = "      0.00"
config.appFormat.fromVar(appVar).toScreenValue() = "      $ 0.00"

But I cannot narrow it further: when I run this code with debugger, it returns "?" instead of "      $ 0.00".

I even inlined the call to fromVar as fillows:
config.appDataPres = ((NumberFormat)appFormat).new NumberBufGui((NumberType) appVar);
with the same effect.

Now I need to know, whether the method fromVar() is incorrect, or it is incorrectly called and another method with another result should have been called. Is there any documentation about this?

**#21 - 05/27/2019 11:43 AM - Hynek Cihlar**

The cause of the problem with the unexpected leading spaces seems to be in NumberType.toString(), which doesn't handle correctly formatting of the leading non-format chars. Beware that I didn't spend enough time to investigate more whether this is the only place, where the format is calculated wrong. Also you may put a break point in the method to see when the display value is being formatted.

**#22 - 06/05/2019 12:05 PM - Alexei Kaigorodov**

I found that if format string starts with ">", then the value string is left aligned, and if it starts with "$" or " ", it is right aligned. 4GL makes visible value string left-aligned in all cases.

NumberType.toString() always returns string with leading spaces. I could not catch the moment when that spaces are removed when 4WD makes left-aligned string.

**#23 - 06/07/2019 06:35 AM - Alexei Kaigorodov**

I found that the decision to print decimal as left or right aligned is taken at the constructor
@   public NumberFormat(String pformat)
...

```
// NOTE: At present we set this flag for limited number of cases,
      // in order to minimize possible regressions, but switching it to true
      // permanently might be also correct.
      if (fs.userLeft.length() > 0 && !rightAligned)
         splitFormat = true;

the value of @splitFormat is later used  as a flag of right alignment.
```

if pformat=="$ >>>,>>9.99" as in hotel gui demo, then fs.userLeft=="$ " and splitFormat = true.

4GL for such pformat still uses left alignment.

I could remove this sentence, do that splitFormat always == false and he walue is always drawn left-aligned in 4GL, but I am not sure this would cause incompatibilities in another places.

Need an advice what to do next.

**#24 - 06/07/2019 07:11 AM - Greg Shah**

Please check how your testcase behaves in ChUI mode.

**#25 - 06/07/2019 07:14 AM - Greg Shah**

To be clear: you should rework the testcase to remove GUI elements such as images.  I think much of the buttons and so forth in the testcase are unnecessary since we really just need to display a fill-in with the correct data and format.  Once you have that minimized so that it will run in ChUI or GUI, I can test it on a ChUI system.

**#26 - 06/07/2019 09:57 AM - Alexei Kaigorodov**

*- File bug3691.p added*

Greg Shah wrote:

> Please check how your testcase behaves in ChUI mode.

it behaves the same way as in Swing mode.

The testcase is minimized. (see uploaded file bug3691.p).

**#27 - 06/10/2019 01:26 PM - Greg Shah**

Your mention of Swing suggests that you tested bug3691.p in FWD. I meant for you to test in 4GL ChUI, not FWD ChUI. Anyway, I have run the test in 4GL ChUI and it does also left align the formatted value.

Please carefully test the program testcases/uast/num_edit.p which has a wide range of numeric format strings. This program should easily run in both ChUI and GUI. I want you to test the following:

- 4GL ChUI (use _progres.exe on Windows, see https://knowledgebase.progress.com/articles/Article/000038172; on Linux OpenEdge the character client is pro)
- FWD ChUI
- 4GL GUI (use prowin32.exe on Windows; on Linux OpenEdge there is no GUI client)
- FWD GUI

Identify the following:

- Any alignment (or other) differences between GUI and ChUI in the 4GL.
- Any differences between 4GL ChUI and FWD ChUI.
- Any differences between 4GL GUI and FWD GUI.

Document all of the results here. From there you will design a fix to resolve all the issues that you find.

**#28 - 06/10/2019 01:32 PM - Greg Shah**

*- File bug3691.p added*

Please do not use the // style comments in programs that are not otherwise required to run in OpenEdge 11.x. v11 is when that style of comments was added to the 4GL, so by adding this usage, it is not possible to run the programs on older versions. Generally we don't want to have such limits.

The attached version has only /* style comments so it runs on older 4GL versions.

**#29 - 06/11/2019 04:49 AM - Alexei Kaigorodov**

*- File num_edit.test.csv added*

Greg Shah wrote:

> Identify the following:
>
> - Any alignment (or other) differences between GUI and ChUI in the 4GL.

1. int "$(zzzzz)": GUI: “$    12 ”, ChUI: “ $   12 ”. (FWD GUI and GHUI follow 4GL GUI).
2. decimal "$>>>>9.99": 4GL ChUI: “$1.23” - max 1 integer digit, 4GL ChUI: “   $123.00” up to 5 integer digits
3. decimal "$(>,>>9.99)": 4GL ChUI: “$1.23” - max 1 integer digit, 4GL ChUI: “$ 1,234.00” up to 4 integer digits

> - Any differences between 4GL ChUI and FWD ChUI.

1. int "$>>>>>": 4GL ChUI: "$12   " left alignment, FWD ChUI: "   $12" right alignment
2. int "$(>>>>>)": 4GL ChUI: "$ 12   " left alignment, FWD ChUI: "  $ 12 " right alignment
3. int "abc->,>>>,>>9": 4GL ChUI: ""abc12" left alignment, FWD ChUI: "       abc12" right alignment
4. decimal "$>>>>9.99": 4GL ChUI: "$1.23" max 1 integer digit, FWD ChUI: "   $12.00" right alignment, up to 5 integer digits
5. decimal "$(>,>>9.99)": 4GL ChUI: "$1.23" max 1 integer digit, FWD ChUI: "   $12.00" right alignment, up to 4 integer digits

- Any differences between 4GL GUI and FWD GUI.

1. int "$>>>>>": 4GL GUI: "$12   " left alignment, FWD GUI: "   $12" right alignment
2. int "$(>>>>>)": 4GL GUI: "$ 12   " left alignment, FWD GUI: "  $ 12 " right alignment, excessive space at the beginning
4. decimal "$>>>>9.99": 4GL GUI: "$12.00" left alignment, FWD GUI: "   $12.00" right alignment
4. decimal "$(>,>>9.99)": 4GL GUI: "$ 12.00" left alignment, FWD GUI: "  $ 12.00 " right alignment, excessive space at the beginning
5. decimal "abc->,>>>.99": 4GL GUI: "abc12.00" left alignment, FWD GUI: "   abc12.00" right alignment

So the most frequent inconsistency is for formats starting with "$" sign: left alignment in 4GL and right alignment in FWD.

**#30 - 06/11/2019 05:19 AM - Alexei Kaigorodov**

*- File num_edit.test.csv added*

the updated test report uploaded

**#31 - 06/13/2019 08:32 AM - Alexei Kaigorodov**

*- File num_edit.test.csv added*

Made a fix for wrong alignment, and ran the test num_edit.p against both FWD Client ChUI and FWD Client GUI. The file with updated testing results is attached.

The only inconsistency with 4GL is now processing the format "abc->,>>>,>>9". For example, the value "12" in this format is shown as "abc12" in 4GL and "abc 12" in FWD - that is, an additional space is inserted. It is inserted in the file p2j/src/com/goldencode/p2j/util/NumberType.java: line 2414. This insertion is commented but I could not understand its sense.

**#32 - 06/13/2019 05:14 PM - Greg Shah**

What branch did you put the fix in?  Create and use 3691a if you have not already done so.

> This insertion is commented but I could not understand its sense.

Text formatting of a numeric value can be done for 2 purposes:

- for display on the screen (as part of a widget displaying a value)
- for usage as a string (which might be written into a report or somehow stored or used)

Please note that there can be differences in how these two cases get formatted.  Perhaps that is what you are seeing here.  You can test the string case using code like this: message string(12345, "abc->>>>9")..  Although this shows the resulting string to the user, it does not use the internal widget formatting to do so.

The screen display is what is used by the widgets in num_edit.p.  That will use the NumberFormat class which may need to implement things differently than NumberType.

**#33 - 06/14/2019 01:13 AM - Alexei Kaigorodov**

The fix for the wrong alignment is comitted to the branch 3691a, revision 11317.

The comment at the line p2j/src/com/goldencode/p2j/util/NumberType.java:2386 reads:
// there is a quirk in the 4GL where the following conditions
// cause the extra blank space to precede any user defined
// text rather than follow it as is normal:
So one may think that of that conditions are met, an extra space should be inserted.
In fact, the following code do the opposite:
if (<<conditions>>)            {
bail = true;
}

```
if (!bail)
              sb.append(negative ? c : ' ');
that is, an extra space is inserted when the conditions are not met.
At first glance, there is an error and if (!bail) should be replaced with if (bail). And indeed, such a change
 fixes the error described in my previous note, but creates a couple of other discrepancies.
```

Besides, there are other problems with this comment:
-  it talks about "extra blank space to precede any user defined text@, but the string buffer which is passed as a parameter already contains user-defined text, so in order to insert extra blank space to precede user defined text, sb.append(negative ? c : ' ') should be replaced with sb.insert(0,negative ? c : ' '), but I am afraid this will cause even more bugs in other places.
- it says // example 1: 12345 using "abc->>>>9" yields " abc12345"
but in fact, 4GL yelds "abc12345".

So now we have to resolve the contradiction between the code and the comment: what is right and what to fix.

**#34 - 06/14/2019 09:54 AM - Greg Shah**

Please look at the 9.1E proghand.pdf (this is one of the documents listed in the "Getting Started with FWD Development". In that book, there is a chapter named "Representing Data". Inside that chapter is a section entitled "Applying Formats with Displaying Widgets". This is one of the few places where the 4GL has documentation about format strings. I think it is worth reading this so that you are aware of the range of possible format string specifications.

Make sure that num_edit.p can be used to handle all format string combinations. If some valid cases are missing, please enhance the 4GL program accordingly.

Analyze the code in question for its affect on the possible format string and value inputs. Then you can prepare changes needed to match the behavior exactly.

Make sure to test the changes with both the string and display usage, since these can differ as I've explained previously. This means that an equivalent to num_edit.p needs to be made (call it num_fmt.p) which uses the string formatting on the same possible format strings, data types and values. This program is nice because it can be completely non-interactive and can store/compare against the expected results as captured in the 4GL. Of course, the same tests for string and display formatting must be compared between the 4GL and FWD, so you must run both programs in both 4GL and FWD.

Report and fix any deviations found.

**#35 - 06/14/2019 10:16 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Please look at the 9.1E proghand.pdf (this is one of the documents listed in the "Getting Started with FWD Development". In that book, there is a chapter named "Representing Data". Inside that chapter is a section entitled "Applying Formats with Displaying Widgets". This is one of the few places where the 4GL has documentation about format strings.

Yes I've read that. I could not find any other place where the meaning of format characters like '>' is explained.

> Make sure that num_edit.p can be used to handle all format string combinations. If some valid cases are missing, please enhance the 4GL program accordingly.

Yes I added 2 cases which are mentioned in the comment.

> Make sure to test the changes with both the string and display usage, since these can differ as I've explained previously. This means that an equivalent to num_edit.p needs to be made (call it num_fmt.p) which uses the string formatting on the same possible format strings, data types and values. This program is nice because it can be completely non-interactive and can store/compare against the expected results as captured in the 4GL. Of course, the same tests for string and display formatting must be compared between the 4GL and FWD, so you must run both programs in both 4GL and FWD.

Is it possible to make these tests automated? that is, is it possible to extract string value from messages and widgets?

**#36 - 06/14/2019 10:22 AM - Constantin Asofiei**

Alexei Kaigorodov wrote:

> Is it possible to make these tests automated?

The SCREEN-VALUE attribute should give you the value:

```
def var ch as char.
def var ch2 as char.
form ch with frame f1.
ch2 = ch:screen-value in frame f1.
// compare this with the expected value
```

Also, there is the string(val, format) function which can be used to format a value.

> that is, is it possible to extract string value from messages and widgets?

How are you using the MESSAGE statement?

**#37 - 06/14/2019 10:24 AM - Greg Shah**

When I say "string formatting", it means use the STRING(val, fmt) to get the result.  When I say "display formatting", it means use the SCREEN-VALUE attribute.

**#38 - 06/14/2019 10:31 AM - Alexei Kaigorodov**

Constantin Asofiei wrote:

> How are you using the MESSAGE statement?

as message string(12345, "abc->>>>9"). So, I need not message statement at all, just get the result of the string() call and compare it with expected value, right?

**#39 - 06/14/2019 11:29 AM - Greg Shah**

Alexei Kaigorodov wrote:

> Constantin Asofiei wrote:
>
> > How are you using the MESSAGE statement?
>
> as message string(12345, "abc->>>>9"). So, I need not message statement at all, just get the result of the string() call and compare it with expected value, right?

Yes, that is it.

**#40 - 06/18/2019 08:40 AM - Alexei Kaigorodov**

I created a simple ABL program to check how Progress executes number formatting:

**#41 - 06/18/2019 08:52 AM - Alexei Kaigorodov**

*- File num_fmt.p added*

I created a simple ABL program to check how Progress executes number formatting num_fmt.p:

```
BLOCK-LEVEL ON ERROR UNDO, THROW.
define variable format as character no-undo.
define variable value as character no-undo.
do while true:
import unformatted format.
import unformatted value.
put string(value, format) skip.
end.
```

it reads 2 lines from stdandard input and writes one line to standard output.
How ever, I could not run it with commands _progres.exe -b num_fmt.p and with prowin.exe -b num_fmt.p because of the error:

- Batch-mode PROGRESS requires a startup procedure. (1144)

What is startup procedure and gow to add it to the num_fmt.p?

another question is, how to create similar ABL program for display formatting?

**#42 - 06/18/2019 09:08 AM - Hynek Cihlar**

Alexei Kaigorodov wrote:

> I created a simple ABL program to check how Progress executes number formatting num_fmt.p:
>
> BLOCK-LEVEL ON ERROR UNDO, THROW.
> define variable format as character no-undo.
> define variable value as character no-undo.
> do while true:
> import unformatted format.
> import unformatted value.
> put string(value, format) skip.
> end.
>
> it reads 2 lines from stdandard input and writes one line to standard output.
> How ever, I could not run it with commands _progres.exe -b num_fmt.p and with prowin.exe -b num_fmt.p because of the error:
>
> - Batch-mode PROGRESS requires a startup procedure. (1144)

I don't think you really want the batch mode. Instead of the -b, use the -p startup parameter:

prowin.exe -p num_fmt.p

For more details, see
https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dpspr%2Fstartup-parameter-descriptions.html%23.

> another question is, how to create similar ABL program for display formatting?

Are you referring to the SCREEN-VALUE feature? If so, you can do something like this.

```
def var fi as decimal format "999.99". /* define your desired format here */
assign fi = 3.14159.
def frame f fi with side-labels.
enable all with frame f.
display fi with frame f.
message fi:screen-value.
wait-for close of this-procedure.
```

**#43 - 06/18/2019 09:44 AM - Alexei Kaigorodov**

Hynek Cihlar wrote:

> For more details, see
> https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dpspr%2Fstartup-parameter-descriptions.html%23.

thank you for the reference. I found the answer there: prowin.exe -p num_fmt.p -b

>> another question is, how to create similar ABL program for display formatting?

> Are you referring to the SCREEN-VALUE feature? If so, you can do something like this.
> def var fi as decimal format "999.99". /* define your desired format here */

I need to pass the format dynamically, through the standard input, as I do it in the num_fmt.p.
Is it possible?

**#44 - 06/18/2019 10:21 AM - Greg Shah**

> I found the answer there: prowin.exe -p num_fmt.p -b

I don't think you need the -b to run the program.  It will still run without a UI.

> BLOCK-LEVEL ON ERROR UNDO, THROW.

I don't think this is needed either.

> I need to pass the format dynamically, through the standard input, as I do it in the num_fmt.p.

Is it possible?

```
define variable format as character no-undo.
define variable value as character no-undo.
```

```
/* this redirects the terminal to a file */
output to "my-output-filename.txt".

/* this is an infinite loop that will end when the import can't read any more data */
/* because the ENDKEY condition will be raised; this is the default behavior */
repeat:
   import unformatted format.
   import unformatted value.

   /* this will output a line to a down frame; iterating the loop will */
   /* cause an implicit DOWN statement which generates the line end */
   display string(value, format) with frame f down.
end.

/* removes the output redirection */
output close.
```

I haven't tested this, but the idea is there.

**#45 - 06/19/2019 07:29 AM - Alexei Kaigorodov**

Greg Shah wrote:

> [...]
>
> I haven't tested this, but the idea is there.

"format" and "value" were rejected as keywords, I changed them to "fmt" and "val".
Then, I changed the type of the variable "val" to int".

then, when the input was fed with
```
zzzzz
0
$>>>>>
12
```

It printed

```
zzzzz
0
$>>>>>
```

that is:
- is it correct to assign a string to int variable val?
- why first line is empty?
- why input values and formatted values was not printed?

**#46 - 06/19/2019 08:53 AM - Alexei Kaigorodov**

I encountered that expression string(0, "zzzzz") returns blank line.
I ran the following test (both with prowin.exe and _progres.exe) :

```
output to "z0.res".
put string(1, "zzzzz") skip.
put string(0, "zzzzz") skip.
put string(10, "zzzzz") skip.
```

it printed

```
    1

   10
```

Can someone explain this?

**#47 - 06/19/2019 09:37 AM - Greg Shah**

> - is it correct to assign a string to int variable val?

No.  But my testcases didn't do that.  Perhaps you are mis-interpreting what this code does: string(value, fmt_string)?  It translates a non-string value into a string using a given format (or the default of the format is not specified).  Please look it up in the 4GL/ABL Reference.

If you are referring to something else, please post your modified version of the testcase.

> - why first line is empty?

Recall my comment from #3691-34:

> Please look at the 9.1E proghand.pdf (this is one of the documents listed in the "Getting Started with FWD Development". In that book, there is a chapter named "Representing Data". Inside that chapter is a section entitled "Applying Formats with Displaying Widgets". This is one of the few places where the 4GL has documentation about format strings. I think it is worth reading this so that you are aware of the range of possible format string specifications.

If you read that chapter, you will find this description of number formatting using z or Z:

"This character is replaced with a digit. If the digit is a leading zero, Z suppresses that digit,  putting a blank in its place."

> - why input values and formatted values was not printed?

Please post your testcase.  My version of the code only tried to output the formatted string version of the number.

**#48 - 06/19/2019 12:27 PM - Alexei Kaigorodov**

Greg Shah wrote:

> "This character is replaced with a digit. If the digit is a leading zero, Z suppresses that digit,  putting a blank in its place."

the rightmost zero is not a leading zero. Zero value cannot be represented with spaces only. Looks like this is a bug in 4gl.

Another question: how to redirect output to standard output stream (System.out in Java)?
If in the test at https://proj.goldencode.com/issues/3691#note-46 I remove the line

```
output to "z0.res",
```

then _progress.exe prints to stdout, but prowin.exe raises an error.

output to TERMINAL raises error "there is no TERMINAL".

**#49 - 06/19/2019 02:20 PM - Greg Shah**

> "This character is replaced with a digit. If the digit is a leading zero, Z suppresses that digit, putting a blank in its place."
>
> the rightmost zero is not a leading zero. Zero value cannot be represented with spaces only. Looks like this is a bug in 4gl.

I don't think this is a bug.  The description in the 4GL docs says "leading zero" which means the leftmost digit(s) not the rightmost digits.  In string(0, "zzzzz"), the leftmost digit is 0 so that would be suppressed with a blank.  Since there is nothing else in the number, there is no output (other than spaces, but I don't recall if there are spaces or not).

> then _progress.exe prints to stdout, but prowin.exe raises an error.
>
> output to TERMINAL raises error "there is no TERMINAL".

If you are running in batch mode, then there is no terminal so it won't work.  This is why I suggested to not use the -b option.  There is no need for it.

Once you get rid of that, you can output to terminal.  In prowin32.exe it will have a kind of text output in a GUI window.

**#50 - 06/20/2019 12:16 AM - Alexei Kaigorodov**

Greg Shah wrote:

> If you are running in batch mode, then there is no terminal so it won't work.  This is why I suggested to not use the -b option.  There is no need for it.
>
> Once you get rid of that, you can output to terminal.  In prowin32.exe it will have a kind of text output in a GUI window.

I need output not to a GUI window but to a file, so that I could compare the results from different programs. So I need batch mode. I need not terminal, I need standard output.

**#51 - 06/20/2019 04:25 AM - Alexei Kaigorodov**

*- File string_fmt.p added*

*- File display_fmt.p added*

So I managed to run display_fmt.p and string_fmt.p (attached) on both prowin32.exe and _progres.exe. Results are equal except that string_fmt.p adds trailing spaces sometimes. I made a java program to compare results which ignores difference in trailing spaces.

Now the main question is: I have display_fmt.p and string_fmt.p converted to java, how can I run it in batch mode locally, without any server, so that it could read from standard input and write to standard output, just like the original programs do when running in prowin32.exe and _progres.exe?

Then, I guess all that programs should already be saved to some repository, preferrably a dedicated one.

**#52 - 06/20/2019 06:26 AM - Greg Shah**

Alexei Kaigorodov wrote:

> Greg Shah wrote:
>
> > If you are running in batch mode, then there is no terminal so it won't work.  This is why I suggested to not use the -b option.  There is no need for it.
> >
> > Once you get rid of that, you can output to terminal.  In prowin32.exe it will have a kind of text output in a GUI window.
>
> I need output not to a GUI window but to a file, so that I could compare the results from different programs. So I need batch mode. I need not

terminal, I need standard output.

You do NOT need batch mode.

As noted in #3691-44, you need this code to output DISPLAY statements to file:

```
/* this redirects the terminal to a file */
output to "my-output-filename.txt".

...

/* close the terminal redirection and allow regular terminal usage again */
output close.
```

The other (better) way to do it is to use something like this:

```
define variable fmt as character no-undo.
define variable valstr as character no-undo.
define variable val as int no-undo.

/* this creates a named stream
define stream rpt.

define frame f val fmt with down.

/* this opens the named stream and attaches it to a file */
output stream rpt to "my-output-filename.txt".

put stream rpt "====" skip.

repeat:

  import unformatted fmt.
  import unformatted valstr.

  val:format = fmt.
  val = INTEGER(valstr).

   /* this will output a line to a down frame; iterating the loop will */
   /* cause an implicit DOWN statement which generates the line end */
   display stream rpt val with frame f down.

end.

/* close the named stream */
output stream rpt close.
```

I don't know why your display_fmt.p uses PUT statements. Everything can be done with DISPLAY in that case. In fact, everything MUST be done with DISPLAY so that you are using a widget to format the value. My testcase from #3691-44 was wrong in that regard because it was still using the string(val, fmt) approach which is by nature, string formatting instead of display/widget formatting. My testcase here fixes that.

It is VERY important to compare the 4GL and FWD output WITHOUT ignoring spaces. The output must be IDENTICAL even in regard to WHITESPACE.

I've also changed the testcase to include both formatted value and format string on the same line, which makes the output easier to understand in my opinion. You can add the string input value if that is useful...

**#53 - 06/20/2019 06:28 AM - Greg Shah**

The string (non-display) version:

```
define variable fmt as character no-undo.
define variable valstr as character no-undo.
define variable val as int no-undo.

define stream rpt.

output stream rpt to "my-output-file.txt".

put stream rpt "====" skip.

repeat:

  import unformatted fmt.
  import unformatted valstr.

  val = INTEGER(valstr).
  put stream rpt fmt valstr string(val, fmt) skip.

end.

output stream rpt close.
```

**#54 - 06/20/2019 08:59 AM - Alexei Kaigorodov**

I use batch mode so that I could save the output to different files.
It is inconvenient to use hardcoded file name as you suppose.
Is it possible that filename ("my-output-file.txt" in your examples) can be passed as a parameter when the program runs with "prowin32.exe" or "_progres.exe"?

**#55 - 06/20/2019 10:16 AM - Alexei Kaigorodov**

*- File test0.zip added*

Greg,
the program for string formatting testing writes result of each microtest in 3 lines: format. input value, and the formatted result.
Can you please make the display testing program use the same output format?
To check format, you can run .bat files from the attached zip archive test0.zip (please correct the paths to your locations of the "prowin32.exe" and "_progres.exe").

I ran both programs for large input (many format and values). They both show several errors. Unfortunately, the error messages have no details. Will have to use binary search to find offending formats and/or values.

**#56 - 06/20/2019 02:13 PM - Greg Shah**

Alexei Kaigorodov wrote:

> I use batch mode so that I could save the output to different files.
> It is inconvenient to use hardcoded file name as you suppose.
> Is it possible that filename ("my-output-file.txt" in your examples) can be passed as a parameter when the program runs with "prowin32.exe" or "_progres.exe"?

The hard coded filename is just example code. If you need a different filename each time, you can write 4GL code to read user input. A simple example is the UPDATE language statement.

The 4GL doesn't really have command line parameters. There is a way, but it is a bit weird. Just create a simple UI to read the value.

**#57 - 06/20/2019 02:19 PM - Greg Shah**

Alexei Kaigorodov wrote:

> Greg,
> the program for string formatting testing writes result of each microtest in 3 lines: format. input value, and the formatted result.
> Can you please make the display testing program use the same output format?

You can add the PUT statements back in if that is needed. It is also possible to do multi-line output using DISPLAY, see the 4GL references.

> To check format, you can run .bat files from the attached zip archive test0.zip (please correct the paths to your locations of the "prowin32.exe" and "_progres.exe").
>
> I ran both programs for large input (many format and values). They both show several errors. Unfortunately, the error messages have no details. Will have to use binary search to find offending formats and/or values.

We care about the errors and can capture them by running statements with NO-ERROR. Then you can look at the ERROR-STATUS system handle to see the error message, error number and so forth. This can be done at the same time as you have access to the formats and input values so there is no need for a search.

I am also concerned with the case where there is no error but FWD calculates the wrong output text. It would be best if your program actually calculates the exact list of these sort of problems. The best way to do that is to capture the actual output in the 4GL and to write code to compare that output to the formatted output in FWD.

Remember we have told you about SCREEN-VALUE which can be used to read the formatted value of a widget without needing to actually output the value to screen or file.

**#58 - 06/21/2019 07:43 AM - Alexei Kaigorodov**

I fixed the errors in the test formatting programs and committed to secure/code/p2j_repo/testcases/.
The tests are in directory uast/format_string. Go to testI and run test* scripts. Then run cmp* scripts. They show that results of string_fmt are the sane on ChUI and GUI engines, and other results greatly differ. The reason of the difference is that uast/format_string/abl/display_fmt.p produces results cut to 11 (ChUI) or 12 (GUI) characters. I have no idea why the results are cut. Can someone look at it?

**#59 - 06/21/2019 08:35 AM - Greg Shah**

> They show that results of string_fmt are the sane on ChUI and GUI engines, and other results greatly differ.

Please provide more details here.

- Are you reporting about the results in the 4GL or in FWD?
- If in the 4GL, how are you running ChUI versus GUI?  Windows Console mode may be different from the UNIX/Linux ChUI and I don't think we have used Windows for testing ChUI at all.  So we don't know how valid such testing is.
- It is not clear what you mean by "other results".

> The reason of the difference is that uast/format_string/abl/display_fmt.p produces results cut to 11 (ChUI) or 12 (GUI) characters. I have no idea why the results are cut. Can someone look at it?

If the same code executed in ChUI vs GUI generates different results, then:

- If this is in the 4GL, then we need to match the behavior.  But please note my concern above about testing ChUI in Windows.  I think we will need someone to run these tests on UNIX/Linux.
- If this is in FWD, then it will depend on whether that matches the 4GL or not.  If not, then there are bugs to fix in FWD.  This would be the continuation of your task.

Which case is this?

Please remove these files from the testcases/uast/ project (they are all artifacts of conversion and should not be saved):

```
uast/format_string/abl/DisplayFmtF.jast
uast/format_string/abl/display_fmt.p.ast
uast/format_string/abl/display_fmt.p.cache
uast/format_string/abl/display_fmt.p.jast
uast/format_string/abl/display_fmt.p.lexer
uast/format_string/abl/display_fmt.p.parser
uast/format_string/abl/display_fmt.p.pphints
uast/format_string/abl/string_fmt.p.ast
uast/format_string/abl/string_fmt.p.cache
uast/format_string/abl/string_fmt.p.jast
uast/format_string/abl/string_fmt.p.lexer
uast/format_string/abl/string_fmt.p.parser
uast/format_string/abl/string_fmt.p.pphints
```

**#60 - 06/21/2019 11:57 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Please provide more details here.
>
> - Are you reporting about the results in the 4GL or in FWD?

In the 4GL. I asked how to run converted program in FWD so that it has access to standard input but had no answer.

> - If in the 4GL, how are you running ChUI versus GUI?

Chui - by C:\Progress\OE116_64\bin_progres.exe, GUI - by C:\Progress\OE116_64\bin\prowin.exe.

> Windows Console mode may be different from the UNIX/Linux ChUI and I don't think we have used Windows for testing ChUI at all. So we don't know how valid such testing is.

I have no access to the Linux inplementation of 4GL.

> - It is not clear what you mean by "other results".

Other results are obtained by execution of uast/format_string/abl/display_fmt.p on ChUI and GUI.

> > The reason of the difference is that uast/format_string/abl/display_fmt.p produces results cut to 11 (ChUI) or 12 (GUI) characters. I have no idea why the results are cut. Can someone look at it?
>
> If the same code executed in ChUI vs GUI generates different results, then:
>
> - If this is in the 4GL, then we need to match the behavior. But please note my concern above about testing ChUI in Windows. I think we will need someone to run these tests on UNIX/Linux.

Yes please find someone who have access to 4GL on Linux.

> - If this is in FWD, then it will depend on whether that matches the 4GL or not. If not, then there are bugs to fix in FWD. This would be the continuation of your task.

Yes, but I still don't know how to run them ob FWD. Looking at converted string_fmt.p, I hope to create an equivalent java program with main() method which does something like:

```
integer val = TypeFactory.integer();
val.assign(new integer(valstr));
fileWriter.write(valueOf(val, fmt));
```

But for converted display_fmt.p, I have no idea.

> Please remove these files from the testcases/uast/ project (they are all artifacts of conversion and should not be saved):

OK.

**#61 - 06/21/2019 02:42 PM - Greg Shah**

> > If this is in FWD, then it will depend on whether that matches the 4GL or not. If not, then there are bugs to fix in FWD. This would be the continuation of your task.
>
> Yes, but I still don't know how to run them ob FWD. Looking at converted string_fmt.p,

You run it the same way you ran bug3691.p in FWD.  And this is the same way you have run other converted programs.

> I hope to create an equivalent java program with main() method which does something like:

Why?  That would defeat the entire point of this.  You MUST run the converted 4GL inside FWD to see how FWD works and whether it is compatible.

As I noted in #3691-59, I prefer if you would capture the output from 4GL and then calculate the report of issues.

Eugenie: Once Alexei has a final set of testcases, please run them on Linux 4GL to capture the results.

**#62 - 06/21/2019 03:03 PM - Eugenie Lyzenko**

Greg Shah wrote:

> ...
> As I noted in #3691-59, I prefer if you would capture the output from 4GL and then calculate the report of issues.
>
> Eugenie: Once Alexei has a final set of testcases, please run them on Linux 4GL to capture the results.

OK.

**#63 - 06/24/2019 09:01 AM - Alexei Kaigorodov**

When talking about 4GL/FWD (in)compatibility, the first observation is that FWD lacks that handy commands  as _progres.exe and prowin.exe.
Agreed that since FWD conversion takes long time, they better be split in conversion and execution parts. But conversion in FWD is not implemented as a command, but as an ant script. And execution parts is not a single program, but user has to launch server first, and then client.
Then, if call to client referes a wrong classname of the converted program, then it tries to execute program referenced in the server configuration. Probably, this is also an incompatibility.

Anyway, I tried to run converted test programs (from directory testcases/uast/format_string/testIfwd) and found, that the output file contains only lines printed before the main loop. This means, that a pipe between FWD process which executes converted user process and outer process does not work. This is another incompatibility between 4GL and FWD.

Then I created  a variant where the test program reads input information from a named disk file and not from standard input (see  directory testcases/uast/format_string/test_ff), and checked it works on 4GL.
Unfortunately, it did not run on FWD because FWD could find required class com.goldencode.testcases.format_string.test_ff.StringFmtFf, though it is clearly seen in testcases/uast/build/lib/testcases.jar.

All additions are pushed up as revision 1875 in tescases project.

**#64 - 06/24/2019 09:31 AM - Eugenie Lyzenko**

Alexei Kaigorodov wrote:

> ...
> All additions are pushed up as revision 1875 in tescases project.

Are the testcases ready to be run on Linux 4GL?

If so please provide the list of the testcases I have to run and any special notes for the output I need to collect.

**#65 - 06/24/2019 09:40 AM - Alexei Kaigorodov**

The testcases themselves are written in ABL, so must be able to run on any 4gl implementation.
I also prepared simple .bat scripts to make it easy to run on Windows.
I did not make .sh scripts because I cannot test them, but I believe they can be easily converted from .bat files.
And, tests are ready for integer values only, not decimal.

**#66 - 06/24/2019 09:59 AM - Eugenie Lyzenko**

Alexei Kaigorodov wrote:

> The testcases themselves are written in ABL, so must be able to run on any 4gl implementation.
> I also prepared simple .bat scripts to make it easy to run on Windows.

> I did not make .sh scripts because I cannot test them, but I believe they can be easily converted from .bat files.
> And, tests are ready for integer values only, not decimal.

Again, I need a list of the Progress 4GL source files to test. I see 2 *.p files in format_string/abl directory. Any more test I've missed?

**#67 - 06/24/2019 10:13 AM - Greg Shah**

> When talking about 4GL/FWD (in)compatibility, the first observation is that FWD lacks that handy commands as _progres.exe and prowin.exe.

FWD is designed to support the conversion and execution of large applications. In such a scenario, running a simple program from the command line is not the approach. This means that there is the conversion and launching of the server before you can run the client driver (our equivalent of the 4GL command line programs).

> Then, if call to client referes a wrong classname of the converted program, then it tries to execute program referenced in the server configuration.
> Probably, this is also an incompatibility.

No, I think you misunderstand what is happening here. The converted program is NEVER run on the client. As discussed in the Runtime Architecture , the converted code only runs on the server. The client is a kind of presentation engine. It is essential that you understand these details.

> Anyway, I tried to run converted test programs (from directory testcases/uast/format_string/testIfwd)

There are no 4GL programs in testcases/uast/format_string/testIfwd/. I don't understand what you are trying to say here.

By the way, the shell scripts in that directory reference ../bin/runComparator.sh which does not exist in testcases/uast/, so that can't work for anyone else on the team.

> Then I created a variant where the test program reads input information from a named disk file and not from standard input (see directory testcases/uast/format_string/test_ff), and checked it works on 4GL.
> Unfortunately, it did not run on FWD because FWD could find required class com.goldencode.testcases.format_string.test_ff.StringFmtFf, though it is clearly seen in testcases/uast/build/lib/testcases.jar.

Are you trying to execute a converted program using the client:cmd-line-option:startup-procedure= command line option? If so, you MUST use ./format_string/test_ff/string_fmt_ff.p instead of the converted Java name.

The only time you use the converted Java class name is when you use p2j-entry in the directory, and in that case you MUST append .execute to the class name for it to work.

Also: I can see that inside this class you are referencing filenames like "feed.csv", which means these files MUST be in your current directory when you start the ClientDriver or they will not be found.

**#68 - 06/24/2019 10:15 AM - Alexei Kaigorodov**

Eugenie Lyzenko wrote:

> Again, I need a list of the Progress 4GL source files to test. I see 2 *.p files in format_string/abl directory. Any more test I've missed?

For now, only these 2 are ready. But they are executed bu different 4gl engines - GnUI and GUI, so 4 tes cases in sum.

**#69 - 06/24/2019 10:19 AM - Greg Shah**

Eugenie Lyzenko wrote:

> Alexei Kaigorodov wrote:
>
> > ...
> > All additions are pushed up as revision 1875 in tescases project.
>
> Are the testcases ready to be run on Linux 4GL?
>
> If so please provide the list of the testcases I have to run and any special notes for the output I need to collect.

Eugenie: Please wait until Alexei has all the testcases finished and has ChUI and GUI results from Windows. At that time you can capture the output on Linux and we can compare the results.

Alexei: I prefer if you have a single 4GL "driver" program that runs any other 4GL code and captures all needed output. Running many different scripts from different directories seems likely to be error prone. As mentioned before, it is better to calculate (in the 4GL) if a result is or isn't correct and write the resulting report out. It makes it less effort to run this multiple times and it eliminates all the extra scripting that is platform dependent. In other words, please write all of this in the 4GL and make a single program that can be run to tell us the answer. Then converting and running this is platform independent and simpler to repeat later.

**#70 - 06/24/2019 10:32 AM - Eugenie Lyzenko**

Greg Shah wrote:

> Eugenie Lyzenko wrote:
>
>> Alexei Kaigorodov wrote:
>>
>>> ...
>>> All additions are pushed up as revision 1875 in tescases project.
>>
>> Are the testcases ready to be run on Linux 4GL?
>>
>> If so please provide the list of the testcases I have to run and any special notes for the output I need to collect.
>
> Eugenie: Please wait until Alexei has all the testcases finished and has ChUI and GUI results from Windows.  At that time you can capture the output on Linux and we can compare the results.

OK.

**#71 - 06/25/2019 07:35 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Alexei: I prefer if you have a single 4GL "driver" program that runs any other 4GL code and captures all needed output.  Running many different scripts from different directories seems likely to be error prone.

I created a prototype 4gl driver in new directory testcases/uast/format_string/driver.
It consists of 2 programs: one generates the file with expected results and the other uses it to check that actual results are correct.

However, it is rejected with diagnostics:

You cannot define streams inside an internal procedure. (3359)

Can you please advise how to get rid of this error?

And, some small number of scrips is needed anyway, like launchers of _progres.exe. Now they are in 2 variants: as .bat and .sh scripts. I am thinking to use some scripting language common for both Linux and Windows. Candidates are Python and Powershell. What do you think?

**#72 - 06/25/2019 04:37 PM - Greg Shah**

> You cannot define streams inside an internal procedure. (3359)
>
> Can you please advise how to get rid of this error?

Move the define stream to the external procedure (anywhere outside of the internal procedure).  Then you can use it in the internal procedure.

**#73 - 06/25/2019 04:51 PM - Greg Shah**

> I created a prototype 4gl driver in new directory testcases/uast/format_string/driver.

It consists of 2 programs: one generates the file with expected results and the other uses it to check that actual results are correct.

Make one 4GL program that calls the others.  Minimize parameters and make it run as simply as possible.

> And, some small number of scrips is needed anyway, like launchers of _progres.exe. Now they are in 2 variants: as .bat and .sh scripts. I am thinking to use some scripting language common for both Linux and Windows. Candidates are Python and Powershell. What do you think?

Why is a script needed?  The driver should simply be executed without any parms.  No other processing should be needed.

We prefer not to encode other logic in random scripting languages unless it is really needed.

**#74 - 06/25/2019 11:08 PM - Alexei Kaigorodov**

Greg Shah wrote:

> Make one 4GL program that calls the others.  Minimize parameters and make it run as simply as possible.

We need to a) take results from 4gl implementation and b) compare them against FWD.

So we need 2 different programs: one takes results and the other takes and compare. How these 2 different tasks can be done in one program, assuming we cannot use parameters?

> Why is a script needed? The driver should simply be executed without any parms.

To simply execute the driver user has to call a command like

```
C:\Progress\OE116_64\bin\prowin.exe -p driver.p -b
```

and even more complex command when executing with FWD runtime.
Scripts are needed to encapsulate this long commands into one short command.

**#75 - 06/25/2019 11:55 PM - Greg Shah**

> To simply execute the driver user has to call a command like
>
> C:\Progress\OE116_64\bin\prowin.exe -p driver.p -b
>
> and even more complex command when executing with FWD runtime.
> Scripts are needed to encapsulate this long commands into one short command.

When executing on different systems there will be different paths and configurations. Why not just describe for the user how to launch it and let them handle it themselves?

As mentioned above, the testcases should not rely upon batch mode to capture output. Please use streams to output to specific file destinations.

**#76 - 06/26/2019 02:07 AM - Greg Shah**

> So we need 2 different programs: one takes results and the other takes and compare.

OK.

You may want to use OPSYS and WINDOW-SYSTEM to generate filenames that match the platform and ChUI/GUI.

**#77 - 06/26/2019 02:33 AM - Alexei Kaigorodov**

Greg Shah wrote:

> When executing on different systems there will be different paths and configurations.  Why not just describe for the user how to launch it and let them handle it themselves?

Sure there must be description how to launch, and example scripts are the best kind of descriptions.

> As mentioned above, the testcases should not rely upon batch mode to capture output.  Please use streams to output to specific file destinations.

I do use streams to named files, both for reading and writing.

A couple of new questions:
- In https://proj.goldencode.com/issues/3691#note-52, you declared a program which prints data formatted using widgets to an output stream. Is it possible to convert it to a function with single parameter which
returns string?
- Is is possible in 4gl to pass a function as a parameter? or better declare an interface and create objects which implements that interface and pass that objects?

**#78 - 06/26/2019 04:16 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Are you trying to execute a converted program using the client:cmd-line-option:startup-procedure= command line option?  If so, you MUST use ./format_string/test_ff/string_fmt_ff.p instead of the converted Java name.

I tried to pass ./format_string/driver_dec/test_dec.p, but server complained:

[06/26/2019 15:09:47 NOVT] (StandardServer.standardEntry:WARNING) {00000006:00000019:bogus} User-specified startup program '/home/avk/Bugfix/3691/testcases/uast/format_string/driver_dec/test_dec.p' not found. Falling back to default.

But the file exists:
avk@msi-leop:~/Bugfix/3691/testcases/uast/format_string/testIfwd$ ls ~~l /home/avk/Bugfix/3691/testcases/uast/format_string/driver_dec/test_dec.p -rw-rw---~~ 1 avk avk 1280 Jun 26 14:21 /home/avk/Bugfix/3691/testcases/uast/format_string/driver_dec/test_dec.p

**#79 - 06/26/2019 04:32 AM - Greg Shah**

> I do use streams to named files, both for reading and writing.

Then drop the -b from your 4GL command line.  If it doesn't work then you have more edits to implement.

> Is it possible to convert it to a function with single parameter which returns string?

Yes, you can try this.  The core technique for getting a widget to format a value can be seen here:

```
def var val as int.
def frame f val.

val:format = "$ zzz99".
val:screen-value = "00042".
message val:screen-value.
```

The frame definition would be done once at the top of the program.  It would not be in the function, but you would reference the widget in the function.

In this example I use message to see the result, but you would not use message.  I'll let you look at the FUNCTION definition in the 4GL references to turn this into a function.

> Is is possible in 4gl to pass a function as a parameter?

No.

> or better declare an interface and create objects which implements that interface and pass that objects?

We are still working on support for the 4GL OO.  It is better to avoid those features for now.

**#80 - 06/26/2019 04:41 AM - Greg Shah**

> But the file exists:
> avk@msi-leop:~/Bugfix/3691/testcases/uast/format_string/testIfwd$ ls l
> /home/avk/Bugfix/3691/testcases/uast/format_string/driver_dec/test_dec.p
> ~~rw-rw-~~ 1 avk avk 1280 Jun 26 14:21 /home/avk/Bugfix/3691/testcases/uast/format_string/driver_dec/test_dec.p

After conversion, this file does not matter.  It is definitely not used in FWD at runtime.

> [06/26/2019 15:09:47 NOVT] (StandardServer.standardEntry:WARNING) {00000006:00000019:bogus} User-specified startup program '/home/avk/Bugfix/3691/testcases/uast/format_string/driver_dec/test_dec.p' not found. Falling back to default.

- Did you convert it?
- Did you jar the result?
- Is the Java class (com.goldencode.testcases.format_string.driver_dec.TestDec.class) in the jar?
- Is there an entry for it in the name_map.xml which is in the jar file?

**#81 - 06/26/2019 04:46 AM - Alexei Kaigorodov**

Greg Shah wrote:

> > I do use streams to named files, both for reading and writing.
>
> Then drop the -b from your 4GL command line.  If it doesn't work then you have more edits to implement.

when the test detects difference between actual and expected values, then in batch mode it simply prints an error and continue to execute, and without -b option it pop-ups a window with the error message and waits until user clicks a button.
I believe our goal is to make huge test base which runs without human intervention and sends emails where list all the test results.

> > Is is possible in 4gl to pass a function as a parameter?
>
> No.

Is it possible in FWD to use dynamic function feature as described in https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dvref/dynamic-function-function.html?

**#82 - 06/26/2019 05:02 AM - Greg Shah**

> pop-ups a window with the error message and waits until user clicks a button.
> I believe our goal is to make huge test base which runs without human intervention and sends emails where list all the test results.

Use NO-ERROR.  Check to 4GL documentation to understand what it does.  Use ERROR-STATUS to see if an error occurred and what the error number and error message are.  This avoids the popup and lets you program the solution.  Do not use batch mode.

> Is it possible in FWD to use dynamic function feature as described

Yes.

**#83 - 06/26/2019 05:16 AM - Alexei Kaigorodov**

Greg Shah wrote:

> - Is the Java class (com.goldencode.testcases.format_string.driver_dec.TestDec.class) in the jar?

no, I made an error in file name to be converted. BTW why FWD converter does not complain about wrong file names?

After filename fixed, java compiler reports errors during convertion:

```
[javac] /home/avk/Bugfix/3691/testcases/uast/src/com/goldencode/testcases/format_string/driver/MakeFeed.java:8
3: error: local variables referenced from a lambda expression must be final or effectively final
    [javac]                valStr.assign(subscript(valstrs, k));
    [javac]                                             ^
    [javac] 6 errors
```

See testcases revision 1878.

4GL runs this .p file successfully. Should I file a bug report?

**#84 - 06/26/2019 05:36 AM - Greg Shah**

> Should I file a bug report?

Sure.  For now, you will have to just rework the code to avoid the issue.

**#85 - 06/27/2019 03:06 AM - Alexei Kaigorodov**

Driver reworked to avoid Extent statement. Now the code compiles but fails at runtime because it cannot find external procedure makeCheck.p, though 4gl executes the code successfully.
Minimized test to demonstrate the bug can be found in /uast/format_string/testFWD/test_int.p, revision 1879.

**#86 - 06/27/2019 07:56 AM - Alexei Kaigorodov**

Greg Shah wrote:

> > Is it possible to convert it to a function with single parameter which returns string?
>
> Yes, you can try this.  The core technique for getting a widget to format a value can be seen here:
>
> [...]
>
> The frame definition would be done once at the top of the program.  It would not be in the function, but you would reference the widget in the function.

I could not find a way to make global frame definition. I made a prototype for that functions, see format_strings/driver/formatters.i, functions named *FrameFormatter. They now return either zero or empty string. Can someone fix them, please?

**#87 - 06/28/2019 12:52 AM - Greg Shah**

*- Related to Bug #4123: conversion generates incorrect java code for Extent statement added*

**#88 - 06/28/2019 03:15 AM - Greg Shah**

It is important that you figure it out so that you can do this in the future as needed.  The def frame can be at the top of the file.  If you need different frames, name them differently.  Or you can put multiple widgets in the same frame and only use the ones you need.

**#89 - 06/28/2019 10:17 AM - Alexei Kaigorodov**

Greg Shah wrote:

> It is important that you figure it out so that you can do this in the future as needed.  The def frame can be at the top of the file.  If you need different frames, name them differently.  Or you can put multiple widgets in the same frame and only use the ones you need.

Implemented (see format_strings/driver/formatters.i, revision 1882).  The frame is declared as a local variable.

That functions sometimes give different results than the string(value, format) functions (different alignment).

**#90 - 06/29/2019 08:29 AM - Alexei Kaigorodov**

It turned out that references to external procedures and include files must be relative to testcases/uast, not to the location of the referencing procedure (which is evident incompatibility with 4gl).

The next FWD problem is that the code converted from

```
if not length(expected) EQ length(actual) OR expected NE actual then DO:
        put formatter "(" fmt ", " valstr ")='" actual "'; but '" expected "' expected." skip.
        errCount = errCount + 1.
        IF errCount = 10 THEN
        LEAVE toplevel.
    END.
```

is executed (I see it under debugger) but prints nothing. With 4GL, it prints messages even when the strings are equal. Guess the reason is that the standard output from server is not redirected to the client, which is an incompatibility.

if put is changed to display, then a window with diagnostic messages appear. But for the purposes of automated testing, we need messages to be intercepted and emailed to interested parties.

**#91 - 06/29/2019 10:12 AM - Constantin Asofiei**

Alexei Kaigorodov wrote:

> It turned out that references to external procedures and include files must be relative to testcases/uast, not to the location of the referencing procedure (which is evident incompatibility with 4gl).

What is the P2J_HOME value when you convert, and what is the working dir for the conversion process?  Something is wrong in your setup, the conversion must start from the testcases/uast folder, and P2J_HOME should be set to ..

> The next FWD problem is that the code converted from ... is executed (I see it under debugger) but prints nothing.

Please point me to the test in the testcases/ project which shows this, I want to take a look.

**#92 - 06/29/2019 10:28 AM - Alexei Kaigorodov**

Constantin Asofiei wrote:

> Alexei Kaigorodov wrote:
>
> > It turned out that references to external procedures and include files must be relative to testcases/uast, not to the location of the referencing procedure (which is evident incompatibility with 4gl).
>
> What is the P2J_HOME value when you convert, and what is the working dir for the conversion process?  Something is wrong in your setup, the conversion must start from the testcases/uast folder, and P2J_HOME should be set to ..

Conversion starts at testcases/uast/, and P2J_HOME is set to the full path of testcases/uast/p2j, where p2j is a symbplic link to the P2J project.

> > The next FWD problem is that the code converted from ... is executed (I see it under debugger) but prints nothing.
>
> Please point me to the test in the testcases/ project which shows this, I want to take a look.

There are 2 test against FWD in testcases/uast/format_strings/driver: test_frame_int_fwd.p and test_string_dec_fwd.p.

**#93 - 06/30/2019 06:25 AM - Greg Shah**

> include files must be relative to testcases/uast, not to the location of the referencing procedure (which is evident incompatibility with 4gl).

This is not a limitation for real applications.  What you were doing is a bad practice and even that could be handled by configuring the FWD runtime propath differently.

> Guess the reason is that the standard output from server is not redirected to the client, which is an incompatibility.

Please refer back to the runtime architecture of FWD.  No output or input are EVER done on the server.  If you debug into the equivalent of PUT in FWD, you will see that it calls the client and passes state.  It is the client that implements the output or input.  You must debug in both places to follow the flow.

> put formatter "(" fmt ", " valstr ")='" actual "'; but '" expected "' expected." skip.
> ...
> if put is changed to display, then a window with diagnostic messages appear. But for the purposes of automated testing, we need messages to be intercepted and emailed to interested parties.

Please STOP using stdout (output to the unnamed stream).  Do NOT use batch mode.  At all.  Not in 4GL and not in FWD.  It is a BAD design.  Please put all output into named streams that are opened against specific files.

> test_frame_int_fwd.p and test_string_dec_fwd.p

Do NOT write 4GL code that is specific to FWD.  This code should be the same for either the 4GL and FWD.  When you run it in the 4GL it should generate results and compare against previously captured results.  When you run it in FWD it generate results and compare against previously captured results.  No difference in the code.

I'll let Constantin offer any other ideas.

**#94 - 06/30/2019 11:19 PM - Alexei Kaigorodov**

Greg Shah wrote:

> > include files must be relative to testcases/uast, not to the location of the referencing procedure (which is evident incompatibility with 4gl).
>
> This is not a limitation for real applications.  What you were doing is a bad practice and even that could be handled by configuring the FWD runtime propath differently.

- what do you call bad practice?
- what should be done instead?

**#95 - 07/01/2019 04:15 AM - Constantin Asofiei**

Alexei, about your issue with include files must be relative to testcases/uast.  The problem is generated by the fact that, when you write your testcases, in 4GL the folder structure is like this:

```
--> project_folder
  --> file1.p
  --> file1.i
```

but in FWD, when you convert them, the folder structure is:

```
--> testcases/uast/
    --> format_string
        --> driver
            --> file1.p
            --> file1.i
```

and the P2J_HOME when converting is set to testcases/uast.

If you modify your 4GL folder structure to be the same, like:

```
--> project_folder
    --> format_string
        --> driver
            --> file1.p
            --> file1.i
```

and then you try to run the program while in project_folder (and NOT in format_string/driver), you will see that 4GL will not find the include file, if is specified like {file1.i}.

So, as Greg mentioned, this can be solved by setting the PROPATH, in both p2j.cfg.xml and directory.xml, to include the format_string/driver folder - this will allow FWD to find the include file and the RUN statement's external program, while having as P2J_HOME the . (current) folder.

The alternative is to make the 4GL folder structure be the same as the one in FWD, and run the program via e.g. pro -p format_string/driver/file1.p - this will require for you to have the RUN and include files relative to current folder, like {format_string/driver/file1.i} and RUN format_string/driver/file1.p.

**#96 - 07/01/2019 05:40 AM - Greg Shah**

> The alternative is to make the 4GL folder structure be the same as the one in FWD, and run the program via e.g. pro -p format_string/driver/file1.p - this will require for you to have the RUN and include files relative to current folder, like {format_string/driver/file1.i} and RUN format_string/driver/file1.p.

This is the correct answer.  Doing it this way is a best practice.  Large 4GL applications work this way.

> what do you call bad practice?

Changing current directory in order to run certain programs is a bad practice.

> what should be done instead?

See above.

**#97 - 07/01/2019 07:24 AM - Alexei Kaigorodov**

Refactored *.p files to use names starting with format_string everywhere. Checked that all they run in 4gl. However, execution in FWD encounters a bug in FWD (reported as https://proj.goldencode.com/issues/4126). Looks like some stale data from previous conversions are used. I removed uast/build and uast/src, it did not help. What else should be cleaned?

**#98 - 07/01/2019 07:45 AM - Alexei Kaigorodov**

thanks to Constantin Asofiei, running uast/clean.ch helped.

Next problem is:
following code in format_string/bin/makeCheck.p

```
DEFINE STREAM err.
output stream err to value(errFileName).
```

where errFileNameis is set to "format_string/generated/feedDec.csv"
causes server error

```
SEVERE: {main} ** "format_string/generated/feedDec.csv" was not found. (293)
```

though this file can br seen from the uast directory:

```
avk@msi-leop:~/Bugfix/3691/testcases/uast$ ls -l format_string/generated/feedDec.csv
-rw-r--r-- 1 avk avk 50922 Jul  1 16:18 format_string/generated/feedDec.csv
```

To reproduce:
- pull revision 1884 of testcases
- cd uast; ./clean.sh
- make conversion and start server
- cd uast/format_string/bin
- run ./testIntChUI.sh

**#99 - 07/01/2019 08:00 AM - Greg Shah**

Your current directory is uast/format_string/bin/ and you are opening a relative path format_string/generated/feedDec.csv which does not exist from that current directory.  The actual filename would be ../generated/feedDec.csv, right?

This should fail in the 4GL too, unless you have set the PROPATH differently.

Again, the solution here is keep your current directory at uast/ and reference everything with names that are relative to that location.  This will work in the 4GL and FWD.

Don't set the PROPATH differently just for this program, that would lead to madness.  In other words, we have thousands of programs in the testcases project.  We can't (and should not) set the PROPATH differently for even a small fraction of these programs.

**#100 - 07/01/2019 08:35 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Your current directory is uast/format_string/bin/

Yes

> Again, the solution here is keep your current directory at uast/

This would require to:
- manually type long commands like format_string/bin/testIntChUI.sh
- or encapsulate that commands in scripts, place them at uast/  and make the number of files in that directory even bigger
- or program changing current directory from uast/format_string/bin/ to uast, which you condemned as  a bad practice.

what variant should I use?

**#101 - 07/01/2019 08:53 AM - Greg Shah**

None of these plans are correct.  The purpose of a driver .p is to run ALL of a given process.  So you can create:

- 1 driver program that does 100% of all capture activity, from the top level current directory using relative names for everything.  This same program would need to run in both 4GL and FWD without any changes.
- 1 driver program that does 100% of all comparison activity, from the top level current directory using relative names for everything.  This same program would need to run in both 4GL and FWD without any changes.

No scripts are needed.  In 4GL, you should be able to run each driver program using something like this:

<path_to_4gl_exe>\<progress_executable> -p <relative_path_to_driver_program>\<driver_program_name>

In FWD, you would use the ClientDriver with the correct parameters for the startup program.

Don't use batch mode, don't use scripts.  The driver programs handle everything.

**#102 - 07/01/2019 09:51 AM - Alexei Kaigorodov**

Greg Shah wrote:

> None of these plans are correct.  The purpose of a driver .p is to run ALL of a given process.

That is, the comparison driver runs all the 8 variants: GUI/ChUI, decimal/integers, strings/frames? It could take some time.

**#103 - 07/01/2019 10:00 AM - Greg Shah**

Alexei Kaigorodov wrote:

> Greg Shah wrote:
>
> > None of these plans are correct.  The purpose of a driver .p is to run ALL of a given process.

> That is, the comparison driver runs all the 8 variants: GUI/ChUI, decimal/integers, strings/frames? It could take some time.

Generally, yes and it doesn't matter if it takes some time.

To capture output from GUI and ChUI are different runs of the same capture driver, right?

**#104 - 07/01/2019 10:05 AM - Alexei Kaigorodov**

Greg Shah wrote:

> To capture output from GUI and ChUI are different runs of the same capture driver, right?

Right, just with different 4gl engines (or FWD client is called with different arguments).

**#105 - 07/01/2019 10:20 AM - Greg Shah**

Alexei Kaigorodov wrote:

> Greg Shah wrote:
>
> > To capture output from GUI and ChUI are different runs of the same capture driver, right?
>
> Right, just with different 4gl engines (or FWD client is called with different arguments).

Good.  Yes, that is the correct way.

**#106 - 07/03/2019 09:46 AM - Alexei Kaigorodov**

Both drivers work. Checker shows that results on 4gl in ChUI and GUI modes are identical. FWD results in ChUI and GUI modes also are identical, but differ from that on 4gl. There are 13 cases of incompatibility:
decFrameFormatter(~~zzz., 99.876)='99.'; but '100.' expected.~~
~~decFrameFormatter(~~**., 99.876)='*99.'; but '100.' expected.**
**decFrameFormatter(~~999., 99.876)='099.'; but '100.' expected.~~**
**~~decFrameFormatter(->>>., 99.876)='99.'; but '100.' expected.~~**
**~~decStringFormatter(-zzz., 99.876)='  99.'; but ' 100.' expected.~~**
**~~decStringFormatter(~~**., 99.876)=' *99.'; but ' 100.' expected.
decStringFormatter(-999., 99.876)=' 099.'; but ' 100.' expected.
decStringFormatter(->>>., 0.12)='  .'; but '    .' expected.
decStringFormatter(->>>., 1.23)=' 1.'; but '   1.' expected.
decStringFormatter(->>>., 23.45)=' 23.'; but '  23.' expected.
decStringFormatter(->>>., 99.876)=' 99.'; but ' 100.' expected.
decStringFormatter(->>>., -1.23)='-1.'; but '  -1.' expected.
decStringFormatter(->>>., -10.45)='-10.'; but ' -10.' expected.

Need to investigate why the incompatibilities found manually earlier, not discovered by this test.

**#107 - 07/04/2019 05:51 AM - Alexei Kaigorodov**

*- File Screenshot at 2019-07-04 16-46-48.png added*

*- File dialog.p added*

Filed a bug report #4130 "FWD truncates decimal part instead of rounding when displaying" which describes the reason of most incompatibilities listed at #3691-106.

As for incompatibilities listed at #3691-29, they are not caused by formatting. In the case when the visible field to display formatted string is wider than that string, it could be aligned to left or right edge. For numerical fields, 4GL and FWD behave differently. A test which demonstrates this is attached, along with a screenshot with both 4GL and FWD executions.
I could not found the rules which govern such alignment and cannot say which implementation is wrong.

**#108 - 07/04/2019 11:01 AM - Greg Shah**

> I could not found the rules which govern such alignment

If you have specific questions, post them here.

> and cannot say which implementation is wrong.

FWD must be wrong since there is a deviation.

**#109 - 07/04/2019 11:26 AM - Alexei Kaigorodov**

I have a question: if a decimal number has more digits after the point than the format allows, then 4gl rounds that number rather than truncate it. Anyone knows where this behavior is specified?

**#110 - 07/04/2019 11:36 AM - Eugenie Lyzenko**

Alexei Kaigorodov wrote:

> I have a question: if a decimal number has more digits after the point than the format allows, then 4gl rounds that number rather than truncate it. Anyone knows where this behavior is specified?

It is better(and faster) to write simple 4GL test and see what is actually happen. Because the 4GL documentation is not always complete/consistent/correct.

**#111 - 07/04/2019 11:37 AM - Greg Shah**

> Anyone knows where this behavior is specified?

Are you asking for where the 4GL documentation explains this (the 4GL reference for "Data Types" may have something useful)?  Or are you asking about where FWD implements it (see the NumberType and decimal classes)?

**#112 - 07/04/2019 11:38 AM - Greg Shah**

But Eugenie's point is correct.  The 4GL docs are incomplete and sometimes wrong.

**#113 - 07/04/2019 11:40 AM - Alexei Kaigorodov**

Greg Shah wrote:

> > Anyone knows where this behavior is specified?
>
> Are you asking for where the 4GL documentation explains this (the 4GL reference for "Data Types" may have something useful)?  Or are you asking about where FWD implements it (see the NumberType and decimal classes)?

I am asking about documentation, but will be grateful for pointing to FWD implementation also.

**#114 - 07/04/2019 01:37 PM - Greg Shah**

My previous comments:

**the 4GL reference for "Data Types" may have something useful**

**see the NumberType and decimal classes**

**#115 - 07/05/2019 06:23 AM - Alexei Kaigorodov**

I have a question (see #3691-107).

The dialog.p contains roomRate2:screen-value = roomRate:screen-value.. But the screenshot shows that
roomRate and roomRate2 looks different. That is, the visible field to display formatted string is wider than that string (screen-value), it could be aligned to left or right edge. For numerical fields, 4GL and FWD behave differently.

Where in the FWD code is alignment chosen to place screen-value attribute at the screen?

**#116 - 07/05/2019 09:43 AM - Greg Shah**

> Where in the FWD code is alignment chosen to place screen-value attribute at the screen?

This was mentioned in #3691-3:

> For widgets, the display formatting (and format strings) are handled by com/goldencode/p2j/ui/client/format/DisplayFormat and its child classes.

You can look inside FillIn and FillInGuiImpl to see how they are used.

**#117 - 07/08/2019 08:59 AM - Alexei Kaigorodov**

Greg Shah wrote:

> You can look inside FillIn and FillInGuiImpl to see how they are used.

Set a lot of breakpoints in FillInGuiImpl, decimal.java and FilleInWdget. They work while the window is initialized bot does not work when I edit the content of the fields. This is strange, as editing field content should include invocation of decimal.toString(String fomat), and othermethods.

**#118 - 07/08/2019 03:46 PM - Greg Shah**

> They work while the window is initialized bot does not work when I edit the content of the fields.

Please explain what you mean by "edit the content".

Have you looked at the DisplayFormat classes?  As I noted in #3691-3 and #3691-116, this is the core class for rendering the text of a value for display purposes.

**#119 - 07/09/2019 02:55 AM - Alexei Kaigorodov**

Greg Shah wrote:

> > They work while the window is initialized bot does not work when I edit the content of the fields.
>
> Please explain what you mean by "edit the content".

look at the attached screenshot Screenshot%20at%202019-07-04%2016-46-48.png
It shows windows with input fields, labelled as Rate for Day. "Edit" means I type characters in that fields with keyboard.

> Have you looked at the DisplayFormat classes?  As I noted in #3691-3 and #3691-116, this is the core class for rendering the text of a value for display purposes.

I set breakpoints on some methods of DisplayFormat and NumberFormat. Debugger stops at these breakpoints when the window is prepared, but does not stops when I edit the input fields.

**#120 - 07/09/2019 05:28 AM - Greg Shah**

> Debugger stops at these breakpoints when the window is prepared, but does not stops when I edit the input fields.

Look deeper that this set of classes.  Look at DisplayFormat.Presentation.

**#121 - 07/09/2019 09:08 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Look deeper that this set of classes.  Look at DisplayFormat.Presentation.

I set breakpoints to almost all methods of Presentation, NumberBuf and CharBuf - no effect.

**#122 - 07/09/2019 09:22 AM - Greg Shah**

Set breakpoints in FillIn.processKeyEvent() and FillIn.draw().  Then follow the event processing flow through the DisplayFormat classes, especially the input() method.

**#123 - 07/09/2019 09:27 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Set breakpoints in FillIn.processKeyEvent() and FillIn.draw().  Then follow the event processing flow through the DisplayFormat classes, especially the input() method.

No effect.
Can it be so that after the application window is initialized, all the work on processing key events is moved to another process?

**#124 - 07/09/2019 09:35 AM - Greg Shah**

No.  Are you debugging the FWD client?

**#125 - 07/09/2019 09:37 AM - Alexei Kaigorodov**

Greg Shah wrote:

> No.  Are you debugging the FWD client?

I an debugging the FWD server.

**#126 - 07/09/2019 09:42 AM - Greg Shah**

Yes, you are looking at the wrong process.  As noted in #3691-67:

> As discussed in the Runtime Architecture, the converted code only runs on the server. The client is a kind of presentation engine. It is essential that you understand these details.

If you are not debugging the presentation engine, you will not see the editing.

**#127 - 07/09/2019 11:42 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Yes, you are looking at the wrong process.  As noted in #3691-67:
>
> > As discussed in the Runtime Architecture, the converted code only runs on the server. The client is a kind of presentation engine. It is essential that you understand these details.
>
> If you are not debugging the presentation engine, you will not see the editing.

So, the code which prepares the input field and code which handles the user input in that fields run in different processes. Interesting approach.

So, about the only remaining problem of this bug: wrong right alignment of numeric fields when format has user-defined prefix, like "$" in "$ >>>,>>9.99".
I propose to change the line 2312 in file NumberFormat.java from:
String result = d.toString(fDef, splitFormat, forceSign);
to
String result = d.toString(fDef, rightAligned, forceSign);

#### #128 - 07/09/2019 02:10 PM - Greg Shah

I am OK with trying the change from splitFormat to rightAligned.  You can put it into branch 4130a.  Does it fix all issues in #3691-29?

Do the existing changes in 4130a fix all the issues in #3691-106?

If there are any other identified issues that are unresolved, please let me know.

#### #129 - 07/11/2019 12:52 AM - Alexei Kaigorodov

Greg Shah wrote:

> I am OK with trying the change from splitFormat to rightAligned.  You can put it into branch 4130a.  Does it fix all issues in #3691-29?

Yes

> Do the existing changes in 4130a fix all the issues in #3691-106?

Yes

> If there are any other identified issues that are unresolved, please let me know.

I discovered that so called floating-decimal display format in string mode works different in FWD and 4gl:
string(>>,>99.99<<<, 0.12)="00.12" in FWD; but "    00.12" on 4gl. That is, left-aligned in FWD and right-aligned in 4gl.

#### #130 - 07/11/2019 10:22 AM - Greg Shah

> I discovered that so called floating-decimal display format in string mode works different in FWD and 4gl:

string(>>,>99.99<<<, 0.12)="00.12" in FWD; but " 00.12" on 4gl. That is, left-aligned in FWD and right-aligned in 4gl.

Please fix it as well.  Use 4130a as the branch.

**#131 - 07/12/2019 08:35 AM - Alexei Kaigorodov**

Greg Shah wrote:

> Please fix it as well.  Use 4130a as the branch.

I found the reason of the wrong alignment:

file NumberType.java line 1215:
if (rightAlign && needed > sbl && (missing > 0 || fmt.indexOf("<")  -1))
// then insert spaces

Condition fmt.indexOf("<")  -1 means "format does not contain '<' ". That is, if format contains '<', spaces are not inserted.

If the condition is changed to:
if (rightAlign && needed > sbl)

then the tests uast/format_string/bin/test*.sh pass for all combinations of formats and data.
However, I'd like to think more about this, in particular, I do not understand the role of variable "missing".

Can anyone explain the role of this variable?

--

**#132 - 07/12/2019 11:02 AM - Greg Shah**

missing > 0 is being used to track the condition where all of the following are true:

- the result must be right aligned
- the format string has a decimal separator
- the format string has < to the right of the decimal separator
- there are fewer digits in the right side numeric value than there are characters to the right of the decimal separator in the format string

> That is, if format contains '<', spaces are not inserted.

Except if missing > 0 which tries to identify an exclusion to the exclusion.  Please check a testcase with the conditions noted above to see if you can recreate it.

**#133 - 07/15/2019 07:55 AM - Alexei Kaigorodov**

I added format and data to recreate a situation when missing>0. This caused an ArrayIndexOutOfBoundsException  in the constructor public NumberBuf(NumberType var). After fixing this, some uast/format_string tests with floating-decimal display format failed: they insert more spaces than 4gl does. Still investigating the reason.

**#134 - 07/16/2019 07:28 AM - Alexei Kaigorodov**

changed the fix to ArrayIndexOutOfBoundsException, now all uast/format_string tests passed. I pushed all fixes related to bugs #3691 and #4130 to /secure/code/p2j_repo/p2j/active/4130a. The latest revision is 11326 "merged with trunk".

**#135 - 07/16/2019 09:05 AM - Greg Shah**

*- Related to Bug #4130: FWD truncates decimal part instead of rounding when displaying added*

**#136 - 07/16/2019 09:07 AM - Greg Shah**

As noted previously, you MUST NOT merge changes from trunk into your branch.  We have a very specific rebasing process which is documented. Please bzr uncommit revision 11326 and instead, rebase the branch using our standard process.

**#137 - 07/16/2019 09:44 AM - Greg Shah**

Code Review Task Branch 4130a Revisions 11324 and 11325

Functionally, I'm OK with trying these changes.

1. Please restore the comments you removed from NumberFormat.Presentation.renderScreen().  Those comments are still valid and seem useful.

2. In NumberType.toString() please fix the code formatting which is not to our standards.

- binary operators must have spaces on both sides
- curly braces are put on their own lines

3. Both NumberFormat and NumberType need history entries.

4. Why did you change gradle/wrapper/gradle-wrapper.properties?

**#138 - 07/16/2019 11:17 AM - Alexei Kaigorodov**

after uncommitting, branch 4130a got into inconsistent state. So I recreate all changes of the branch 4130a in the branch 3691a, taking into account notes from previous code review, and pushed to the branch 4130a. Now it is at revision 11322 "history entries added".
Please review.

**#139 - 07/16/2019 12:52 PM - Greg Shah**

Code Review Task Branch 4130a Revision 11321 and 11322

I've fixed the issues in items 1 and 2 as revision 11323.

1. The code change in decimal is missing.

2. There are code formatting issues.  The "spaces in binary operators" issue was still there and one of the history entries was too long (98 characters is the limit).

3. You started with the wrong trunk revision 11320.  The latest trunk revision is 11326 so this branch needs a rebase.

**#140 - 07/16/2019 12:52 PM - Greg Shah**

I am going to rebase 4130a now.  Do not make any changes to 4130a.

**#141 - 07/16/2019 01:33 PM - Greg Shah**

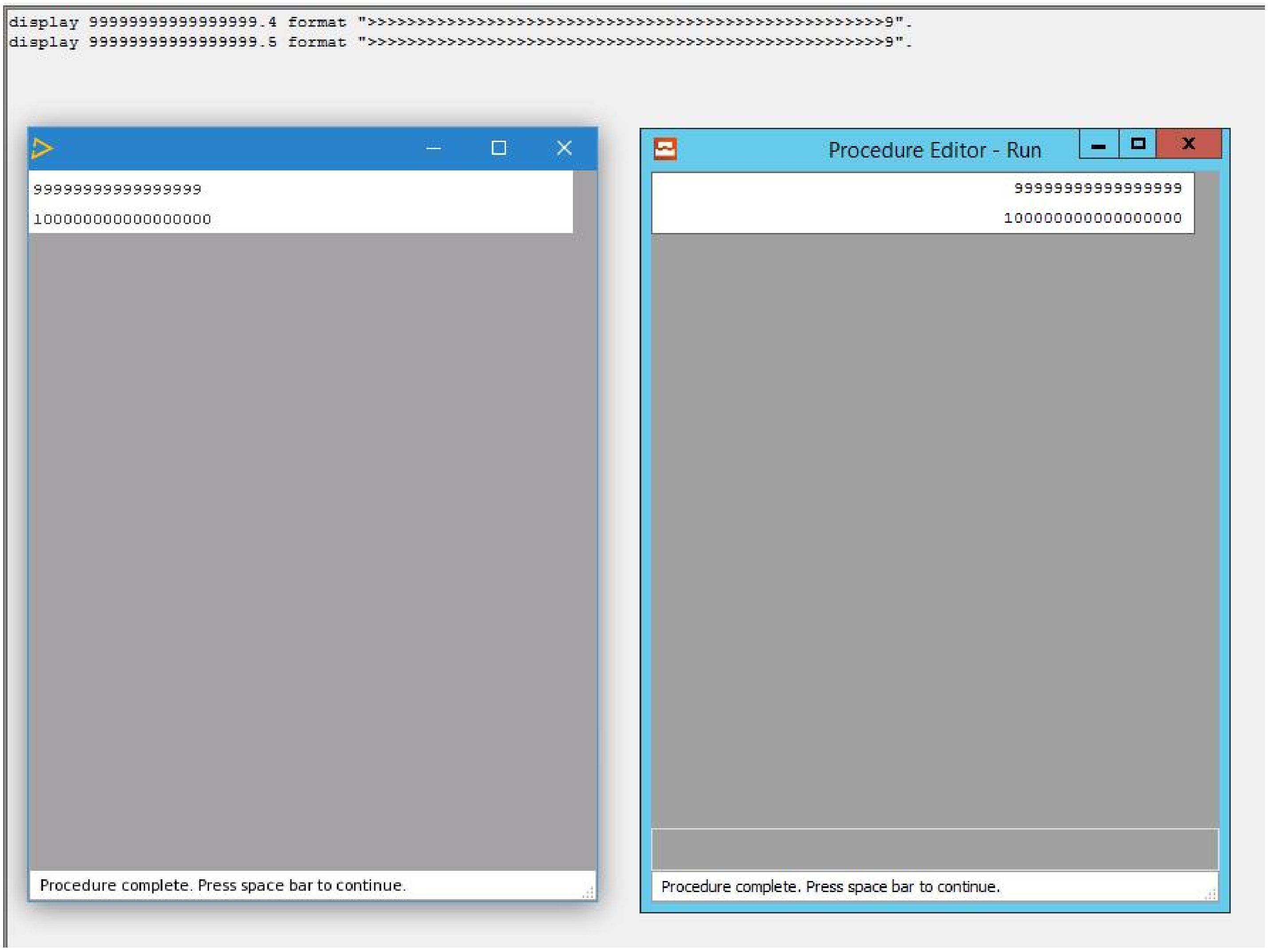
Task branch 4130a was rebased from trunk rev 11326.  The latest revision is now 11327.

**#142 - 08/08/2019 05:22 PM - Greg Shah**

*- File fwd-progress-alignment-difference.png added*

Reported by Vladimir:

> the same code outputs right-aligned in FWD swing, and left-aligned in Progress:



```
display 99999999999999999.4 format ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>9".
display 99999999999999999.5 format ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>9".
```

```
99999999999999999
100000000000000000
```

Procedure complete. Press space bar to continue.

Procedure Editor - Run

```
                                        99999999999999999
                                       100000000000000000
```

Procedure complete. Press space bar to continue.

This duplicates our existing findings in #3691-22.

**#143 - 08/08/2019 05:22 PM - Greg Shah**

There are some fixes in 4130a for this issue.  However, I ran the ChUI regression testing and a quick look a the results showed that the changes broke many of the reports generated using redirected terminal language features.  So the changes are not completely correct right now.  I have not had time to look deeper.

## Files

| | | | |
|---|---|---|---|
| hotel_gui_trunk_rev_11281_format_string_literal_in_wrong_place_20180821.png | 50.3 KB | 08/21/2018 | Greg Shah |
| bug3691_Dollar_Sign.p | 3.02 KB | 05/13/2019 | Alexei Kaigorodov |
| UpdateRateTest.jpg | 50.5 KB | 05/13/2019 | Alexei Kaigorodov |
| Fixed_FWD.png | 19 KB | 05/13/2019 | Alexei Kaigorodov |
| Fixed_4GL.png | 4.88 KB | 05/13/2019 | Alexei Kaigorodov |
| bug3691.p | 1.77 KB | 06/07/2019 | Alexei Kaigorodov |
| bug3691.p | 1.78 KB | 06/10/2019 | Greg Shah |
| num_edit.test.csv | 3.88 KB | 06/11/2019 | Alexei Kaigorodov |
| num_edit.test.csv | 3.9 KB | 06/11/2019 | Alexei Kaigorodov |
| num_edit.test.csv | 4.93 KB | 06/13/2019 | Alexei Kaigorodov |
| num_fmt.p | 244 Bytes | 06/18/2019 | Alexei Kaigorodov |
| display_fmt.p | 490 Bytes | 06/20/2019 | Alexei Kaigorodov |
| string_fmt.p | 343 Bytes | 06/20/2019 | Alexei Kaigorodov |
| test0.zip | 1.65 KB | 06/20/2019 | Alexei Kaigorodov |
| dialog.p | 857 Bytes | 07/04/2019 | Alexei Kaigorodov |
| Screenshot at 2019-07-04 16-46-48.png | 342 KB | 07/04/2019 | Alexei Kaigorodov |
| fwd-progress-alignment-difference.png | 16.2 KB | 08/08/2019 | Greg Shah |