User Interface - Bug #2809

Bug # 2677 (New): fix drawing and functional differences between P2J GUI and 4GL GUI

implement missing behaviour related to HIDDEN attribute

11/03/2015 06:04 AM - Constantin Asofiei

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	70%
Category:		**Estimated time:**	0.00 hour
Target version:	Cleanup and Stabilization for GUI		
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 11/03/2015 06:13 AM - Constantin Asofiei

For frames and windows, setting the HIDDEN attribute to true will block drawing it until an explicit statement which may show it is executed directly on it. From 4GL docs for the HIDDEN attribute:

```
Setting the HIDDEN attribute to TRUE prevents the widget from being displayed implicitly.
For a field-level widget, child frame, or child window, this means that the widget is not
automatically made visible when the containing frame or parent window becomes visible. The
widget does not appear unless one of the following occurs:
• It is forced to receive user input (for example, using a SET or PAUSE statement).
• It is explicitly displayed using a VIEW statement or by setting its VISIBLE attribute to
TRUE.

Any action that explicitly displays the widget also resets the HIDDEN attribute to FALSE. If
the widget is already visible, setting its HIDDEN attribute to TRUE makes that widget and any
widgets it parents (and their descendants) invisible (VISIBLE is set to FALSE). The default
value of the HIDDEN attribute is FALSE for all widgets.

In windows, setting the HIDDEN attribute to TRUE prevents implicit display of the hidden
window when you:
• Invoke DISPLAY, ENABLE, and VIEW statements for frames of the window
• View an ancestor or descendant window of the hidden window

This limits flashing side effects caused during set up of the application user interface. In
windows, this attribute is not supported in character mode.
For frames and dialog boxes, setting the HIDDEN attribute to TRUE prevents implicit display
of the frame or dialog box when you invoke DISPLAY or ENABLE statements for the widget
or its descendant frames. This allows the frame or dialog box to remain invisible during actions
that set it up. The HIDDEN attribute has no effect on DISPLAY statements directed to a file,
pipe, or printer.

Note:
Setting a frame or field-level widget’s VISIBLE attribute to TRUE also displays any
parent or ancestor frames, even if their HIDDEN attributes are set to TRUE (resetting
the HIDDEN attributes, if necessary). However, setting a window’s VISIBLE attribute
to TRUE only displays the window if there are no ancestor windows with their
HIDDEN attribute set to TRUE. In any case, the window’s own HIDDEN attribute is
set to FALSE.

For field-level widgets and frames parented by other frames, setting the HIDDEN attribute to
TRUE prevents implicit display of the field-level widget or child frame when its containing
frame or dialog box is displayed. If the frame or dialog box containing the widget is visible,
setting HIDDEN to FALSE for the widget makes the widget visible (the VISIBLE attribute is
set to TRUE). If the containing frame or dialog box is not visible, setting HIDDEN to FALSE
has no effect on the VISIBLE attribute of the widget.

Note:
The HIDE statement sets the VISIBLE attribute for the widget to FALSE. It only sets
```

```
the HIDDEN attribute to TRUE if you hide a field-level widget or child frame whose
containing frame is still visible
```

Areas which need support (in priority order):

1. make HIDDEN work for non-nested frames (I think this is the main reason the GUI code is flashing widgets on the default-window).
2. nested frames: what happens if nested/parent frame is hidden and parent/child frame is then displayed/set
3. make HIDDEN work for non-hierarchy windows
4. work with window hierarchy (again, parent/child hidden or not, use UI statements on hidden parent/child)
5. DIALOG-BOX
6. frame

#2 - 11/06/2015 12:31 PM - Constantin Asofiei

2677a revision 11014 address these:

- HIDDEN attribute for widget/frame is reported correctly throughout the test suite
- DISPLAY is a no-op if the current or any parent frame is HIDDEN
- when drawing something on a HIDDEN window (or if any of its ancestors is hidden), do not show the window

The testcases/uast/hidden folder contains tests related to HIDDEN attribute for widget/frame. A description about each test encoding meaning can be found in hidden_attribute_main.i. The HIDDEN attribute is reported correctly with rev 11014, but there are some issues related to drawing:

- hidden_attribute.p (always = true, which means HIDDEN is assigned regardless of value - true or false)
  - 1, 2, 3a, 4a, 5a, 6, 7at, 7bt, 7ct, 7dt, 7f, 8at, 8bt, 8ct, 8dt, 8f, 9, 10, 11t, 11f: ok
  - 3b-5,6,7,8; 4b-5,6,7,8; 5b-5,6,7,8: these are tests with SET/PROMPT-FOR/UPDATE for a child frame - first GO shows the frame, second discards it (it should be only one GO and frame f2 - the child - never drawn)
- hidden_attribute2.p (always = false, which means HIDDEN is assigned only if is set to true)
  - 1a, 2, 3, 4a, 4b, 5, 6, 7at, 7f, 8at, 8bt, 8ct, 8dt, 8f, 9, 10, 11t, 11f: ok
  - 1b: this tests VIEW statement, for a child frame - f2. P2J does not draw both f1 and f2, when VIEW f2 (the inner frame) is executed, regardless of frame f1 (the parent frame)'s state. The main issue here is this: GenericFrame internal APIs need to distinguish between a real VIEW statement call and a simulated VIEW part of some other statement (i.e. DISPLAY, PROMPT-FOR, etc)
  - 7bt, 7ct, 7dt: these test the VISIBLE attribute for a parent frame's widget or a child frame (f2). In P2J the frames f1,f2 are never drawn, when something other than the root frame (f1) has its VISIBLE attribute turned on - we might need to introduce some recursivity in some points, to go up the hierarchy and force a VIEW for all the parents - might be related to the VIEW problem above.
- 3b,4b,5b - these test SET/PROMPT-FOR/UPDATE for a child frame: if both frames (parent and child) are hidden, in some cases the WAIT-FOR is not executed (to wait for a GO)
- common issue - ChUI test only - frame f2 title's is drawn in the incorrect place

#3 - 11/06/2015 12:34 PM - Constantin Asofiei

And something else to mention: hidden_attribute.p and hidden_attribute2.p are ChUI only due to a PUT SCREEN statement - they can't be ran in P2J GUI.

#4 - 11/06/2015 01:23 PM - Greg Shah

What are the next steps for this? Does it make sense to finish all of this off? Can/should some items be deferred?

#5 - 11/06/2015 01:36 PM - Constantin Asofiei

Greg Shah wrote:

> Another approach is to add some code in GenericFrame at the beginning of each API which is associated with a 4GL statement to store which 4GL statement is executed, and set it to null afterwards; when an internal view() is executed, it would see that a 4GL statement is already being executed and do nothing; otherwise, it would set GenericFrame into VIEW mode - this would save us the refactoring time.

Another key issue here is using the same frame in an UI statement, from with a trigger, while i.e. an UPDATE/SET/etc is running... this will require a save/restore mechanism, to allow such nested usage.

#6 - 11/06/2015 02:00 PM - Greg Shah

Somehow, it looks like this task history entry was lost:

Issue #2809 has been updated by Constantin Asofiei.

Greg Shah wrote:

> What are the next steps for this? Does it make sense to finish all of this off? Can/should some items be deferred?

My current concerns are the issues related to VIEW and VISIBLE attribute targeting a child frame (or widget). But this requires some refactoring in GenericFrame to move the internal VIEW code in workers so that a GenericFrame.view() call can be marked/treated as a VIEW statement, and not some internal P2J call. Another approach is to add some code in GenericFrame at the beginning of each API which is associated with a 4GL statement to store which 4GL statement is executed, and set it to null afterwards; when an internal view() is executed, it would see that a 4GL statement is already being executed and do nothing; otherwise, it would set GenericFrame into VIEW mode - this would save us the refactoring time.

The other found issues can wait.

Also, the WINDOW:HIDDEN was not thoroughly tested yet - we need to test combinations of UI statements + MESSAGE/PAUSE/STATUS + misc WINDOW:HIDDEN states (with child or root windows). Something similar is needed for DIALOG-BOX, too.
My plan would be this:

1. spend some time to check if the idea related to GenericFrame/UI statements above works (avoiding refactoring) - if so, implement it.
2. implement the WINDOW:HIDDEN and DIALOG-BOX:HIDDEN; for WINDOW it should not take too long (1-2 days); DIALOG-BOX might behave the same. The issue here is that automation to cover as many cases as possible (as I've done with frames and widgets) might not be possible, but I'll try to think of something.

#7 - 11/06/2015 02:00 PM - Greg Shah

I like the plan.

#8 - 11/11/2015 01:43 PM - Greg Shah

Does the change in 2677a rev 11033 to WindowWidget have any implications in ChUI?

#9 - 11/11/2015 01:48 PM - Constantin Asofiei

Greg Shah wrote:

> Does the change in 2677a rev 11033 to WindowWidget have any implications in ChUI?

No.

#10 - 11/12/2015 01:30 PM - Constantin Asofiei

- % Done changed from 0 to 70

With 2677a rev 11041, testcases/uast/hidden/hidden_attribute_run.p executes OK for all statements. Please review.

Also, some other issues were solved:

1. a bug for resolving the window (if set, CURRENT-WINDOW needs to be used regardless of its hidden/visible state)
2. labels show warning that they "do not fit" only if their text is changed.
3. general support of resolving a frame's window in TC.viewWorker, when there are nested frames.

Note that general support for WINDOW:HIDDEN and DIALOG-BOX:HIDDEN is not added yet. I'll postpone this a little and move into the missing nested frames/frame family changes which directly affect the GUI screen.

#11 - 11/12/2015 02:27 PM - Greg Shah

Code Review Task Branch 2677a Revision 11041

This seems like a good step forward. I did make some minor header and doc cleanups (checked in as rev 11043). I think the use of AspectJ in this case is smart.

1. The GenericFrame.resetStatement() doesn't need a parameter. It seems like the code in UIStatementsAspect could only call resetStatement() when pop is true. This makes the GenericFrame code simpler and the API is more intuitive. See item 2 for my example code.

2. The code in UIStatementsAspect is very repetitive. Why not create a worker method like this:

```
   private Object statementWorker(ProceedingJoinPoint thisJoinPoint, UIStatement stmt)
   throws Throwable
   {
      GenericFrame frame = (GenericFrame) thisJoinPoint.getThis();

      boolean pop = frame.setStatement(stmt);

      try
      {
         return thisJoinPoint.proceed();
```

```
      }

      finally
      {
         if (pop)
         {
            frame.resetStatement();
         }
      }
   }
```

Then just call this from each of the actual aspects.

3. In GenericFrame.makeFrameEnable(), the conditional code protected by if (!frame.config.hidden) sets frame.config().hidden = false;. Aren't frame.config.hidden and @frame.config().hidden the same? And isn't it already false?

#12 - 11/12/2015 02:32 PM - Constantin Asofiei

Greg Shah wrote:

> 3. In GenericFrame.makeFrameEnable(), the conditional code protected by if (!frame.config.hidden) sets frame.config().hidden = false;. Aren't frame.config.hidden and @frame.config().hidden the same? And isn't it already false?

Yes, they are the same. I'll change it to make it consistent.

I'll fix all these later today.

#13 - 11/12/2015 04:34 PM - Constantin Asofiei

I have the changes from issue 11 ready - they are safe, and can be committed now.

#14 - 11/12/2015 04:38 PM - Greg Shah

Go ahead.

#15 - 11/12/2015 04:39 PM - Constantin Asofiei

Greg Shah wrote:

> Go ahead.

Committed to 2677a rev 11047.

#16 - 11/27/2015 07:01 AM - Constantin Asofiei

2677b rev 10975 fixes implicit window show, when is targeted by a frame; the UI statements showing the window are these (for the root frame), assuming the window (and non of its parents) is hidden:

- pristine frame (hidden=visible=no): display, update, set, prompt-for, view, enable
- hidden=no, visible=yes: display, update, set, prompt-for, view, enable
- hidden=yes, visible=no: update, set, prompt-for, view

#17 - 11/27/2015 09:20 AM - Greg Shah

Is there more work still needed for WINDOW:HIDDEN and DIALOG-BOX:HIDDEN?

#18 - 11/27/2015 09:22 AM - Constantin Asofiei

Greg Shah wrote:

> Is there more work still needed for WINDOW:HIDDEN and DIALOG-BOX:HIDDEN?

Yes, I haven't tested yet DIALOG-BOX - and this can't be worked easily until the #2875 (and input freeze problems) are solved.

#19 - 01/25/2016 09:12 AM - Greg Shah

Can you provide an estimate of how much work is left here?

#20 - 01/25/2016 04:15 PM - Constantin Asofiei

Greg Shah wrote:

> Can you provide an estimate of how much work is left here?

What I need is to check how DIALOG:HIDDEN is related to its parent window (and up the hierarchy), what happens if its parent gets hidden/shown while the dialog is active (or if nested dialog-boxes are active). I expect ~1 day of work.

#21 - 03/23/2016 05:05 PM - Greg Shah

- Target version changed from Milestone 12 to Milestone 16

#22 - 05/03/2016 10:12 AM - Constantin Asofiei

Another issue of interest, to check: how setting the HIDDEN/VISIBLE attribute for a WINDOW affects the containing frames (and their widgets). A finding in #3093 suggests that when making a window non-visible, the VISIBLE attribute is reported as FALSE for all its UI tree. But if the window is made visible again, the VISIBLE attribute is restored for the UI tree - it might be just an issue of reporting the attribute? What if the frame is moved from a non-visible window to a visible window?

#23 - 11/16/2016 12:23 PM - Greg Shah

- Target version changed from Milestone 16 to Cleanup and Stabilization for GUI