User Interface - Bug #6053

Widget realisation

02/08/2022 08:52 AM - Marian Edu

Status:	Review	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Marian Edu	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 02/08/2022 09:02 AM - Marian Edu

The widget 'realisation' is somehow inconsistent in FWD, there are two attributes in widget configuration with a big TODO - realized and wasRealized.

- the realize method is not always used
- not all attributes that when accessed causes 'realisation' use the canAccess method that take care of it (with error handling) - screen-value being one of them
- some methods directly sets the realized flag to true while realize method uses wasRealized
- _isRealised uses the realised attribute
- both realised and wasRealised are accessed in many places

Maybe it could be a good idea to clean this up before it grows in a larger task...

#2 - 02/08/2022 09:05 AM - Greg Shah

- Assignee set to Marian Edu

- Start date deleted (02/08/2022)

Yes, please do.

#3 - 04/21/2022 07:44 AM - Marian Edu

- % Done changed from 0 to 70

- Status changed from New to WIP

#4 - 06/10/2022 08:05 AM - Marian Edu

- Status changed from WIP to Review

- File 6053.patch added

- % Done changed from 70 to 100

Attached the patch to remove the temporary solution with wasRealized, we've made tests for widget properties that required realization or can not be changed afterwards in testcases:

- ui/fill_in/attr-force-realize.p

- ui/fill_in/attr-realized-ro.p
- ui/fill_in/attr-require-realize.p
- ui/browse/attr-realized-ro.p
- ui/combo/attr-force-realize.p
- ui/combo/attr-require-realize.p
- ui/radio_set/attr-realized-ro.p
- ui/radio_set/attr-require-realize.p
- ui/selection_list/attr-force-realize.p
- ui/selection_list/attr-realized-ro.p
- ui/selection_list/attr-require-realize.p

#5 - 06/10/2022 09:51 AM - Greg Shah

Hynek: Please review and merge if OK.

#6 - 06/10/2022 10:46 AM - Marian Edu

I think last time when we've tried to remove wasRealized Roger reported some regression on <customer_app_2>. We've tried to test as much as possible in <customer_app_1> and couldn't find any issues but maybe Roger could help with a smoke test there as well.

Thanks

#7 - 06/10/2022 03:03 PM - Hynek Cihlar

Code review 6053.patch.

There are two references to wasRealized causing compilation errors. I ran several test cases with these removed, assuming the simple removal was a correct resolution.

A potentially problematic place in AbstractWidget._setVisible: container.getContentPane().normalizeZOrder(); will be not called when canAccess will be executed for the widget for the widget. Also config.visible won't be assigned.

Another potential issue in AbstractWidget.afterConfigUpdateBase:

```
      if (!c.realized && c.visible && c.visible != beforeUpdate.visible)
      {
         moveToTop();
      }
```

If canAccess is executed for the widget, c.realized will be set to true and moveToTop won't be called.

Another one in ComboBoxGuiImpl.updateSize, the condition if (!config.realized) won't be executed when canAccess is executed for the widget.

And perhaps other similar cases as the above exist.

I'm not saying the above are real issues, just that the change will affect how the mentioned code segments will be executed. Ideally all the places where realized is checked for false (for the subjected widget types - combo box, radio set, selection list and perhaps browse column) should be tested.

#8 - 06/13/2022 03:09 AM - Marian Edu

Hynek Cihlar wrote:

> Code review 6053.patch.
>
> There are two references to wasRealized causing compilation errors. I ran several test cases with these removed, assuming the simple removal was a correct resolution.

The patch was made agains rev #13959 which was the latest one available on xfer when I've created the patch. I can't tell if simple removal was the correct solution, you might have applied the patch for frame adjustment where wasRealized was added in the interim - in that case removal is ok since it had a check on isRealized as well. If someone can push latest revision in xref I can recreate the patch just to be sure.

> A potentially problematic place in AbstractWidget._setVisible: container.getContentPane().normalizeZOrder(); will be not called when canAccess will be executed for the widget for the widget. Also config.visible won't be assigned.

Hynek, canAccess is on GenericWidget - the server-side (model) of the widget and the AbstractWidget is - I think - the real client side implementation. The configuration seems to be passed around in both ways, if changes are done from code then it goes from GenericWidget to AbstractWidget and when changed from UI actions the other way around.

Let aside the fact that canAccess was there before and there are 50+ places where it is called out of which only 3 more were added in the patch, _setVisible is called already from setVisible in afterConfigUpdateBase (see you next comment) and the z-order doesn't really change unless the widget was not realized using display/enable so I have no idea why the z-order normalisation of the container's content pane is done there.

> Another potential issue in AbstractWidget.afterConfigUpdateBase:
> [...]
> If canAccess is executed for the widget, c.realized will be set to true and moveToTop won't be called.

Not sure if this is possible to get in the same 'configuration update' changes for both realize and visible, realize on GenericWidget doesn't pushAttr so maybe in this case just to be safe we can check if the widget changed from hidden to visible and was not realized **before** just in case realize was changed to true in the same attributes push call.

> Another one in ComboBoxGuiImpl.updateSize, the condition if (!config.realized) won't be executed when canAccess is executed for the widget.

I would say whatever extra processing needs to be done based on config.realized should be done in realize method but in this particular case the method does receive the 'before update' configuration so if need be this can be used instead to figure out if the widget was not realized before.

> And perhaps other similar cases as the above exist.
> I'm not saying the above are real issues, just that the change will affect how the mentioned code segments will be executed. Ideally all the places where realized is checked for false (for the subjected widget types - combo box, radio set, selection list and perhaps browse column) should be tested.

Ideally (imho) the configuration should be make private to begin with and move everything that pertains to widget realization in the realize method which is there to be used I guess and don't scatter realization logic all over the places, patch tape has it's use but eventually time come to clean things up :)

If you can push the latest revision to xfer I can recreate the patch today.

Thank

#9 - 06/13/2022 04:10 AM - Hynek Cihlar

Marian Edu wrote:

> If you can push the latest revision to xfer I can recreate the patch today.

Will do, what's the target directory?

I'm looking at the cases I mentioned above in more detail, whether these are real issues.

#10 - 06/13/2022 04:15 AM - Marian Edu

Hynek Cihlar wrote:

> Marian Edu wrote:
>
> > If you can push the latest revision to xfer I can recreate the patch today.
>
> Will do, what's the target directory?

It's not a directory, we get that from bazaar - xfer.goldencode.com/opt/fwd/3821c/ only that I do think you guys are working on an internal development server and that one is usually ahead of xfer.

The revision I've pulled now from it is #13968, is that ok to recreate the patch?

#11 - 06/13/2022 04:36 AM - Hynek Cihlar

Marian Edu wrote:

> The revision I've pulled now from it is #13968, is that ok to recreate the patch?

Yes, that is the latest of the main repository, too.

#12 - 06/13/2022 07:14 AM - Marian Edu

- File 6053.v2.patch added

Hynek Cihlar wrote:

> Marian Edu wrote:
>
> > The revision I've pulled now from it is #13968, is that ok to recreate the patch?
>
> Yes, that is the latest of the main repository, too.

Attached the new patch, did fixed the check on afterConfigUpdateBase as there it was easy to see if the widget was realized before.

I think that there should be only one place where that realized configuration property should be set - either GenericWidget or AbstractWidget should handle the realization and push that info around as needed. Right now realize method on GenericWidget only set that property to true but there was no real realization - the UI implementation isn't even aware the widget was marked as realized. The UI implementation does realize the widget (well, set the property to true) when the widget became visible but that is also pushed from the server-side (GenericWidget) unless I'm missing something :(

#13 - 06/14/2022 04:58 AM - Hynek Cihlar

Marian Edu wrote:

> Hynek Cihlar wrote:
>
> > Code review 6053.patch.
> >
> > There are two references to wasRealized causing compilation errors. I ran several test cases with these removed, assuming the simple removal was a correct resolution.
>
> The patch was made agains rev #13959 which was the latest one available on xfer when I've created the patch. I can't tell if simple removal was the correct solution, you might have applied the patch for frame adjustment where wasRealized was added in the interim - in that case removal is ok since it had a check on isRealized as well. If someone can push latest revision in xref I can recreate the patch just to be sure.
>
> > A potentially problematic place in AbstractWidget._setVisible: container.getContentPane().normalizeZOrder(); will be not called when canAccess will be executed for the widget for the widget. Also config.visible won't be assigned.

Hynek, canAccess is on GenericWidget - the server-side (model) of the widget and the AbstractWidget is - I think - the real client side implementation. The configuration seems to be passed around in both ways, if changes are done from code then it goes from GenericWidget to AbstractWidget and when changed from UI actions the other way around.

Actually currently the realized config field is serialized only from client to server, but not from server to client. On server the field is simply assigned, but not marked to be pushed to client with setAttr. This effectively isolates its state on client from server. I.e. on server a widget is already marked as serialized but client still sees an unrealized state. Is this intentional? If so it will be safer to keep the server state out of config and rather make it part of the server widget itself.

#15 - 07/04/2022 05:45 AM - Marian Edu

Hynek Cihlar wrote:

> Actually currently the realized config field is serialized only from client to server, but not from server to client. On server the field is simply assigned, but not marked to be pushed to client with setAttr. This effectively isolates its state on client from server. I.e. on server a widget is already marked as serialized but client still sees an unrealized state. Is this intentional? If so it will be safer to keep the server state out of config and rather make it part of the server widget itself.

I don't know if this is intentional or not (do hope the question was not for me though) but the more I look at it the clear it becomes there is **no need for wasRealized attribute and not even for realized** - a widget is automatically 'realized' when placed in a field-group, aka: frame or dialog. There is really no need for realize method or anything like that, everything should be done when the frame is set for the widget - either statically through frame definition or dynamically by setting the frame attribute.

So, basically if the frame is set then the widget is fully 'realized' - the frame is really the only requirement here.

#16 - 07/04/2022 07:13 AM - Hynek Cihlar

Marian Edu wrote:

> So, basically if the frame is set then the widget is fully 'realized' - the frame is really the only requirement here.

Marian, you may be right here.

Igor, do you remember the test cases for which you needed the canAccess method and wasRealized flag?

#17 - 07/04/2022 08:42 AM - Greg Shah

> basically if the frame is set then the widget is fully 'realized' - the frame is really the only requirement here.

What about static frames (which have the frame set at compilation time)?

#18 - 07/04/2022 08:50 AM - Hynek Cihlar

Greg Shah wrote:

> basically if the frame is set then the widget is fully 'realized' - the frame is really the only requirement here.

What about static frames (which have the frame set at compilation time)?

Referencing a widget attribute or method on a variable or a field (without it being part of a frame) gives a compile time error.

#19 - 07/04/2022 08:54 AM - Greg Shah

> Referencing a widget attribute or method on a variable or a field (without it being part of a frame) gives a compile time error.

I think this can only happen for dynamic widgets. Static widgets in a static frame (e.g. DEF FRAME, FORM, DISPLAY .. WITH FRAME...) must always know their frame. But these can be unrealized too, if I recall correctly.

#20 - 07/04/2022 08:58 AM - Marian Edu

Greg Shah wrote:

> basically if the frame is set then the widget is fully 'realized' - the frame is really the only requirement here.

What about static frames (which have the frame set at compilation time)?

All attributes that require the widget to be 'realized' can be accessed without error if the widget is placed in a frame - through static frame definition. There is no difference on how the widget's parent/container is set, the errors only occurs for dynamic widget because those doesn't have the container set statically. There is no 'static widget' without a frame, a variable can only have one widget attached if present in a static frame definition.

#21 - 07/04/2022 09:07 AM - Greg Shah

> There is no 'static widget' without a frame, a variable can only have one widget attached if present in a static frame definition.

There are DEF IMAGE, DEF BUTTON, DEF RECTANGLE, DEF MENU, DEF SUB-MENU and DEF BROWSE.

#22 - 07/04/2022 09:20 AM - Hynek Cihlar

Greg Shah wrote:

> > There is no 'static widget' without a frame, a variable can only have one widget attached if present in a static frame definition.
>
> There are DEF IMAGE, DEF BUTTON, DEF RECTANGLE, DEF MENU, DEF SUB-MENU and DEF BROWSE.

Even for these in 4GL a compile time error is shown when referencing widget attributes or methods with no frame assigned.

#23 - 07/04/2022 09:21 AM - Marian Edu

Greg Shah wrote:

> I think this can only happen for dynamic widgets. Static widgets in a static frame (e.g. DEF FRAME, FORM, DISPLAY .. WITH FRAME...) must always know their frame. But these can be unrealized too, if I recall correctly.

Not sure what you mean by 'unrealized', once the frame or parent attribute is set for a widget it can not be changed not even set to unknown. Hence the funny hijack effect of display/update statements that looks like moving the widget from one frame to another.

Other than that if the widget's frame is dynamic then when the frame is deleted the widget is not 'unrealized' and left lingering around without a parent but it is simply deleted - note the widget is also dynamic since it is not possible to place a 'static widget' in a dynamic frame because there is no such a thing as a static widget with no frame.

#24 - 07/04/2022 09:33 AM - Marian Edu

Greg Shah wrote:

> There is no 'static widget' without a frame, a variable can only have one widget attached if present in a static frame definition.

There are DEF IMAGE, DEF BUTTON, DEF RECTANGLE, DEF MENU, DEF SUB-MENU and DEF BROWSE.

The only exception is the MENU that can be used as MENU-BAR for window or POPUP-MENU for other widgets that supports it. So this doesn't even have a frame attribute but owner but I expect if any attribute require this to be realized to actually mean the owner must have been set.

#25 - 07/18/2022 02:54 AM - Hynek Cihlar

A relevant info, #6523-29.

#26 - 07/21/2022 11:54 AM - Igor Skornyakov

Hynek Cihlar wrote:

> Igor, do you remember the test cases for which you needed the canAccess method and wasRealized flag?

Sorry Hynek,
I do remember that I had a battle with wasRealized flag. But it was many years ago and I've completely forgot the details.

Files

6053.patch	23 KB	06/10/2022	Marian Edu
6053.v2.patch	24.9 KB	06/13/2022	Marian Edu