User Interface - Feature #3282

create a 4GL language extension to control screen refresh

04/27/2017 02:24 PM - Greg Shah

Status:	Closed	**Start date:**	04/27/2017
Priority:	Normal	**Due date:**	
Assignee:	Hynek Cihlar	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 04/27/2017 02:25 PM - Greg Shah

Per #3257-59, this task is to create a new 4GL syntax extension (specific to FWD) to disable all drawing to a given window and to re-enable updates (triggering a refresh).

#2 - 06/23/2017 05:39 AM - Hynek Cihlar

For the language extension, would it be alright to introduce new attribute that would disable/enable window refresh?

```
CREATE WINDOW h.
h:DISABLE-REDRAW = TRUE.
/* window redraw is disabled */
h:DISABLE-REDRAW = FALSE.
/* window redraw is enabled */
```

According to #3257 note 59 the customer currently uses WM_SETREDRAW Win32 window message to enable/disable window redraw. I will run some experiments to find out the semantics of this message in order to figure out how this feature currently behaves in respect to the following points:

- should a window redraw (from its buffer) when it is invalidated with another window
- do the window decorations redraw? (icons, system menu, etc.)
- may the window change its state (minimized, maximized, etc.)
- does the window post input messages when redraw disabled?

#3 - 06/23/2017 06:21 AM - Greg Shah

> would it be alright to introduce new attribute that would disable/enable window refresh

Yes, the proposed approach is acceptable.

#4 - 06/23/2017 11:19 AM - Hynek Cihlar

Hynek Cihlar wrote:

> According to #3257 note 59 the customer currently uses WM_SETREDRAW Win32 window message to enable/disable window redraw. I will run some experiments to find out the semantics of this message in order to figure out how this feature currently behaves in respect to the following points:
>
> - should a window redraw (from its buffer) when it is invalidated with another window
> - do the window decorations redraw? (icons, system menu, etc.)
> - may the window change its state (minimized, maximized, etc.)
> - does the window post input messages when redraw disabled?

I took the customer's code that disables/enables window redraw and ran few experiments with it on a demo 4gl app.

Here are the results:

- Sending WM_SETREDRAW message to WINDOW:HWND only affects the "client" portion of the 4gl window (that is the window less all system decorations, less the 4gl status area and less the 4gl message area. Obviously the WINDOW:HWND doesn't resolve to the whole 4gl window but just the window area that may contain 4gl widgets.
- Due to the above, when window redraw is disabled its state can be changed (minimized, maximized, etc., even programatically) and the system decorations redraw.
- When window is redraw disabled, the 4gl status and message areas redraw without any limitations.
- A redraw-disabled window doesn't process mouse input when the input happens in the "client" portion of the 4gl window.
- A redraw-disabled window does process keyboard input, even in the "client" portion of the 4gl window. The visual changes get apparent once the window redraw is enabled again.
- A redraw-disabled window doesn't redraw its "client" area when it is invalidated by another window.
- 4gl widgets that draw in separate system windows (menus, combo-box selection popups, tooltips) do draw and process user input without any limitations when their top-level window parent is redraw-disabled.

#5 - 06/29/2017 11:12 AM - Hynek Cihlar

The implementation has been committed to 3282a and is ready for review. GUI regression tests passed, ChUI tests in progress.

#6 - 07/21/2017 02:10 PM - Greg Shah

Code Review Task Branch 3282a Revision 11156

1. The gap analysis marking in rules/gaps/lang_stmts.rules will not work properly. The change should be in the methods and attributes section of rules/gaps/expressions.rules since this is a new attribute, not a language statement.

When you move the marking there, please put a comment at the end of the line explaining this:

```
<!-- FWD EXTENSION, not a real 4GL attribute -->
```

2. I think there should be a setDisableRedraw(boolean) version which is what would be used most frequently.

3. TopLevelWindow has no real changes, I think the history entry can be backed out.

4. In WindowWorkspace.draw(), why not call isRepaintDisabled() instead of having the code inline. The only difference in result is that the window null check is done before the !isVisible() check. If that is important, we need a comment there.

#7 - 08/17/2017 03:12 AM - Hynek Cihlar

Rebased 3282a against current trunk.

#8 - 08/17/2017 03:58 AM - Hynek Cihlar

The points from the review resolved in 3282a. Also 3282a was merged in 3284a. Please review.

#9 - 09/01/2017 01:32 PM - Greg Shah

Can this task be closed?

#10 - 10/02/2017 08:30 AM - Hynek Cihlar

Code changes with the DISABLE-REDRAW implementation were delivered in task branch 3284b and merged to trunk as revision 11172.

#11 - 10/02/2017 08:32 AM - Hynek Cihlar

Greg Shah wrote:

> Can this task be closed?

I'm planning yet do update the wiki documentation, after that I will mark this task as done.

#12 - 10/04/2017 04:18 PM - Hynek Cihlar

Wiki updated at Screen Refresh Control.

#13 - 10/04/2017 04:19 PM - Hynek Cihlar

- % Done changed from 0 to 100

#14 - 10/04/2017 04:26 PM - Greg Shah

Please add a couple of examples of the 4GL code that is replaced by this approach. The first two items in #3257-59 show this. The first case is bracketed calls to LockWindowUpdate() and the second case is a call to SendMessageA() which is used to send a WM_SETREDRAW (11 or 0x0B) message to disable updates + on re-enabling, the HWND is used as the 1st parameter for a RedrawWindow() call to force the window to repaint.

By showing these cases, the reader can understand how they are use this feature to eliminate the native WIN32 HWND usage. That should be

explained as part of the doc.

You should also explain that no automated conversion is provided for these cases.

Otherwise the doc is good.

#15 - 10/06/2017 07:09 AM - Hynek Cihlar

Greg Shah wrote:

> By showing these cases, the reader can understand how they are use this feature to eliminate the native WIN32 HWND usage. That should be explained as part of the doc.
>
> You should also explain that no automated conversion is provided for these cases.

Done.

#16 - 10/06/2017 07:28 AM - Greg Shah

- Status changed from New to Closed

- Assignee set to Hynek Cihlar

I made some minor edits and I think it is good to go.

#17 - 10/27/2017 11:30 AM - Hynek Cihlar

Extended DSIABLE-REDRAW to all widget types. Previously only WINDOW widget was supported. I checked in the change into newly created task branch 3282b.

Please review.

#18 - 10/28/2017 10:30 AM - Hynek Cihlar

3282b rebased against trunk revision 11184.

#19 - 10/28/2017 10:19 PM - Greg Shah

Code Review Task Branch 3282b Revision 11184

I'm generally fine with the changes.

One question: What is the reason that Window.isRepaintDisabled() is hard coded to false?

#20 - 10/29/2017 12:35 PM - Hynek Cihlar

Greg Shah wrote:

One question: What is the reason that Window.isRepaintDisabled() is hard coded to false?

This ensures that anything in the WINDOW widget tree except window workspace (title bar, message and status area) will redraw even when the WINDOW redraw is disabled. I will add this to the javadoc so that it is clear.

#21 - 10/29/2017 11:32 PM - Greg Shah

Code Review Task Branch 3282b Revision 11186

I'm fine with the code.

#22 - 10/30/2017 02:46 AM - Hynek Cihlar

I checked in a fix of the regression of combo box and menu not handling mouse input in 3282b and also addressed the points from review there. The branch passed ChUI and GUI regression tests. Please review.

#23 - 10/30/2017 07:30 AM - Greg Shah

Code Review Task Branch 3282b Revision 11187

The code is good.

Please rebase. There will be some minor conflicts. Let me know if you think any of those are risky enough to require another round of testing. Otherwise you can merge to trunk.

#24 - 10/30/2017 10:32 AM - Hynek Cihlar

3282b merged to trunk as revision 11186 and archived.