Base Language - Bug #3020

Add support for the IMMEDIATE-DISPLAY and MULTITASKING-INTERVAL SEEION attributes

03/08/2016 02:38 PM - Igor Skornyakov

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

Description

History

#1 - 03/08/2016 02:42 PM - Igor Skornyakov

- File immediate-display.p added

- File immediate-display.mkv added

In a simplest scenario when MULTITASKING-INTERVAL == 0 (default value) the IMMEDIATE-DISPLAY is false (default value) **some** output operation (screen updates of frames made visible with VIEW/ENABLE and messages in the message area) are on hold until it is true or a statement block for input (including PAUSE) operation. Moreover all pending updates (those which are made by the code but are not still visible) are blocked including those ones initiated before IMMEDIATE-DISPLAY was set to false. After the IMMEDIATE-DISPLAY was set to true or a a statement blocks for input all the updates become visible. The error message resumes the output (all pending changes become visible). Contrary to what is said in the Progress documentation the DISPLAY statement is not affected, I've not noticed any impact of the IMMEDIATE-DISPLAY if MULTITASKING-INTERVAL != 0. See the attached immediate-display.p and the immediate-display.mkv screencast.
Stream I/O is not affected by the IMMEDIATE-DISPLAY value.
At least the following should be checked:

1. Will opening of a new window (and close of the existing one) be on hold?
2. Can you move, resize, minimize, maximize 4GL windows while this is active?

The current P2J behaviour looks like a Progress one with MULTITASKING-INTERVAL == 1 (or other small positive number) - in this situation I've not noticed any effect of the IMMEDIATE-DISPLAY as mentioned above.
In any case the changes regarding IMMEDIATE-DISPLAY/MULTITASKING-INTERVAL will change the existing visual behaviour (even in situations when IMMEDIATE-DISPLAY/MULTITASKING-INTERVAL where not assigned).

#2 - 11/28/2016 05:12 PM - Hynek Cihlar

The following findings came to live due to the performance considerations of #3111 (starting at note 237).

According to my testing SESSION:IMMEDIATE-DISPLAY affects how 4GL dispatches enqueued Win32 messages. When SESSION:IMMEDIATE-DISPLAY is set to false the Win32 messages are processed only at the input-processing statements like WAIT-FOR and PAUSE. When SESSION:IMMEDIATE-DISPLAY is set to true, the messages are dispatched on more regular basis. I am not sure how often, it could be after every instruction or every instruction potentially affecting the UI.

4GL draws the GUI with native Win32 UI controls and its own painting routines. It combines these two approaches together: For example, the FILL-IN widget is drawn with native Win32 editor widget, but it is augmented with the 4GL painting routines (for example the widget's background set with BGCOLOR). Another example is a root FRAME widget. Frame widget and its box is drawn directly by 4GL runtime, while the window background behind the frame is the native Win32 window. Another example may be the BROWSE widget. In read-only mode it is drawn entirely by 4GL runtime, but when editing is enabled, it uses the Win32 native editors.

The native Win32 controls repaint themselves only when the host process dispatches messages in the Win32 message queue. And so between the input processing statements (like WAIT-FOR and PAUSE) and while SESSION:IMMEDIATE-DISPLAY set to false only the parts of UI drawn directly by 4GL runtime will repaint. When the time between the 4GL input processing instructions becomes long, the inconsistencies in the UI will become apparent.

It seems that SESSION:IMMEDIATE-DISPLAY only affects the draw behavior of the native Win32 controls, the UI drawn directly by 4GL runtime is not affected by the attribute.

In the following sample one can see how the native 4GL UI responds during a lengthy operation when IMMEDIATE-DISPLAY is set to FALSE. Changing the location of the FILL-IN widget will make its background to repaint correctly while the widget's content is missing. And more, the OS will mark the window as "Not responding" after a couple of seconds (when a user input in the window is attempted) until the PAUSE statement is hit. This may confirm the assumption that 4GL runtime in this case is not dispatching Win32 messages.

```
DEF VAR j AS CHAR.
DEF FRAME f j WITH SIDE-LABELS SIZE 60 BY 20.
DEF VAR i AS INTEGER.
DEF VAR k AS INTEGER.
DEF VAR l AS INTEGER.

VIEW FRAME f.

j:SCREEN-VALUE = "X".
j:BGCOLOR = 3.

SESSION:IMMEDIATE-DISPLAY = FALSE.

DO i = 1 TO 100:
   j:COL = RANDOM(1, 40).
   j:ROW = RANDOM(1, 10).
   j:SCREEN-VALUE = STRING(i).

   /* delay - burn some CPU cycles */
   DO k = 1 TO 1000000:
      l = RANDOM(1, 10).
   END.
END.

PAUSE.
```

At the moment P2J behaves according to the native 4GL with SESSION:IMMEDIATE-DISPLAY set to TRUE. And so may appear sluggish (especially with complex UIs) when compared to native 4GL with SESSION:IMMEDIATE-DISPLAY set to FALSE.

How far do we want to replicate the draw behavior of SESSION:IMMEDIATE-DISPLAY set to FALSE? There are cases where this behavior makes the UI inconsistent (as seen in the example above), on the other hand there are also more legitimate cases, like a BROWSE widget being updated during a lengthy operation (remember BROWSE widget is not a native Win32 control, but is drawn by 4GL runtime directly).

#4 - 09/20/2021 12:54 PM - Greg Shah

- Status changed from New to Test

- % Done changed from 0 to 100

- Start date deleted (03/08/2016)

- Assignee set to Constantin Asofiei

The changes are in task branch 3821c revision 12957.

#5 - 09/20/2021 01:06 PM - Constantin Asofiei

Gaps are updated in 3821c/12958.

Files

immediate-display.mkv	302 KB	03/08/2016	Igor Skornyakov
immediate-display.p	1.58 KB	03/08/2016	Igor Skornyakov