User Interface - Bug #2736

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

fix GUI menu drawing problems in menu/simple_sm.p

09/23/2015 03:36 PM - Greg Shah

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Vadim Gindin	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	GUI Support for a Complex ADM2 App		
billable:	No	**case_num:**	
vendor_id:	GCD		

Description

Related issues:

Related to User Interface - Bug #2707: menu/simple_sm.p causes an infinite lo...	**Closed**

History

#1 - 09/23/2015 03:37 PM - Greg Shah

See #2707-11 for details.

#2 - 10/12/2015 04:57 PM - Greg Shah

- Assignee set to Vadim Gindin

#3 - 10/14/2015 02:23 PM - Greg Shah

Put your changes into task branch 1811s.

#4 - 10/23/2015 05:07 PM - Vadim Gindin

Quote from the #2707

```
run simple_sm.p open the second sub-menu and move mouse pointer to it's body and than out of it: you will see,
 that menu background around the body is drawn as menubar background. There is also some blinking, when moving
 mouse in the sub-menu body between different menu-items.
```

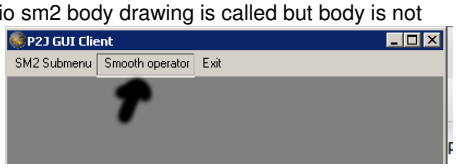
This bug is not reproduced at this moment. But the other is founded: when mouse pointer is moved from menu-item outside of the sub-menu body and return back: focus is not returned.

#5 - 10/27/2015 07:59 AM - Vadim Gindin

- File body_drawing.png added

I'm currently working on fix of the following bug. Assume we have menubar with 2 sub-menus (lets name them sm1 and sm2 correspondingly) at the

first level. When we press sm1 (it's body become opened) and move mouse pointer to sm2, sm2 body must also must become opened, but it doesn't happen.

I've made some changes (revno 10967, branch 2677a) and faced with that bug: after described scenario sm2 body drawing is called but body is not drawn: only thin line of body is drawn under sub-menu title instead of the body (some visual effect) see 

Could you have a look there and advice me why it happen?

#6 - 10/29/2015 11:29 AM - Greg Shah

> When we press sm1 (it's body become opened) and move mouse pointer to sm2, sm2 body must also must become opened, but it doesn't happen.

Was this working previously?

In regard to your changes in revision 10967, are they safe to include in the trunk?

The changes look OK to me, but I need you to update the history entry for each of the files you changed.

#7 - 10/29/2015 12:44 PM - Vadim Gindin

Greg Shah wrote:

> > When we press sm1 (it's body become opened) and move mouse pointer to sm2, sm2 body must also must become opened, but it doesn't happen.
>
> Was this working previously?

I made that earlier, but after some moment those changes were lost. I.e. "pressed" state didn't transferred at all, I've tried to fix it.

> In regard to your changes in revision 10967, are they safe to include in the trunk?

The are safe in regards to other functionality.

#8 - 10/29/2015 06:16 PM - Greg Shah

Please make the history entry updates that are missing.

#9 - 10/30/2015 04:13 AM - Vadim Gindin

Done. revno 10985.

#10 - 11/03/2015 03:23 AM - Vadim Gindin

Does PaintPrimitives contain absolute coordinates (x,y) or relative?

I'm recalling, that current problem is the following. Let's remember "pressed" state transfering scenario: 2 sub-menus in a MENUBAR. User opens the first one's body (by mouse click on title) and moves mouse pointer to the second sub-menu's title and that sub-menu's body must be also opened (without explicit mouse click).

This does not occur. There is some drawing affect (have a look at image in the note 5). May be the body is overdrawn. I've debugged it and founded that sub-menu's code drawing the body is called and the code gd.fill3DRect(0, 0, bodyDim.width, bodyDim.height, true); is also called and PaintPrimitives is processed in SwingEmulatedWindow.draw with the body dimensions:

```
PaintPrimitives.FILL_3D_RECT  x = 0; y = 0; width = 91; height = 46; raised = true
```

Could you advice me where to look the reason?

#11 - 11/03/2015 04:57 AM - Constantin Asofiei

Vadim Gindin wrote:

> Does PaintPrimitives contain absolute coordinates (x,y) or relative?

Coordinates passed to PaintPrimitives are always relative to the translated origin. So if you want to use absolute coordinates, you must use the drawFromOrigin API.

> I'm recalling, that current problem is the following. Let's remember "pressed" state transfering scenario: 2 sub-menus in a MENUBAR. User opens the first one's body (by mouse click on title) and moves mouse pointer to the second sub-menu's title and that sub-menu's body must be also opened (without explicit mouse click).
>
> This does not occur. There is some drawing affect (have a look at image in the note 5). May be the body is overdrawn. I've debugged it and founded that sub-menu's code drawing the body is called and the code gd.fill3DRect(0, 0, bodyDim.width, bodyDim.height, true); is also called and PaintPrimitives is processed in SwingEmulatedWindow.draw with the body dimensions:
> [...]
>
> Could you advice me where to look the reason?

Debug the PaintEvent's being raised against the coordinates used during drawing. This usually happens because there is some drawing code which clips the drawing area so that the menu's body is excluded (if you are drawing both the menu's title and the body in the same draw API call). Another reason might be that the body is overwritten by the workspace - if the body is not on top (z-order), then the workspace will draw last and will overwrite the menu's body.

Also, if nothing works, you can debug the PaintEvent's being posted (via some System.out.println code in TitledWindow.processEvent with the event's source and rectangle) and check if there is one posted for the menu's body.

#12 - 11/04/2015 01:29 PM - Vadim Gindin

Constantin Asofiei wrote:

> Debug the PaintEvent's being raised against the coordinates used during drawing. This usually happens because there is some drawing code which clips the drawing area so that the menu's body is excluded (if you are drawing both the menu's title and the body in the same draw API call). Another reason might be that the body is overwritten by the workspace - if the body is not on top (z-order), then the workspace will draw last and will overwrite the menu's body.
>
> Also, if nothing works, you can debug the PaintEvent's being posted (via some System.out.println code in TitledWindow.processEvent with the event's source and rectangle) and check if there is one posted for the menu's body.

I've tried all proposed variants without success.

1. Clipping. There are 3 different gd.draw(..) calls: title, body, children and the last calls do not use clipping (clip is null). Could the clipping take place somewhere outside of sub-menu?
2. Z-order. Workspace rectangle is always filled before it's children being drawn, see BorderedPanelGuiImpl.draw().
3. PaintEvent's. There was some inaccuracy, but after I fixed it (just for try if it is the source reason) - nothing changed.

#13 - 11/04/2015 04:09 PM - Vadim Gindin

About PaintEvent rectangle.

SubMenuGuiImpl.prepParentLocation() for sub-menu "Smooth operator" adds the height of sub-menu title. It leads to incorrect PaintEvent rectangle during repaint: it's top=69, but it should be about top=48. I.e. for sub-menu in MENUBAR we shouldn't need to increase parent location. I'm not sure if it is correct, but it generates correct PaintEvent after I commented the lines in SubMenuGuiImpl.prepParentLocation():

```
         int fontHeight = FontManager.getFontHeight(wnd, gf.resolveFontNum(), gd);
         p.y += fontHeight + 5 + 5;  // under the title
```

Anyway It didn't solve the main problem.

#14 - 11/05/2015 02:58 AM - Vadim Gindin

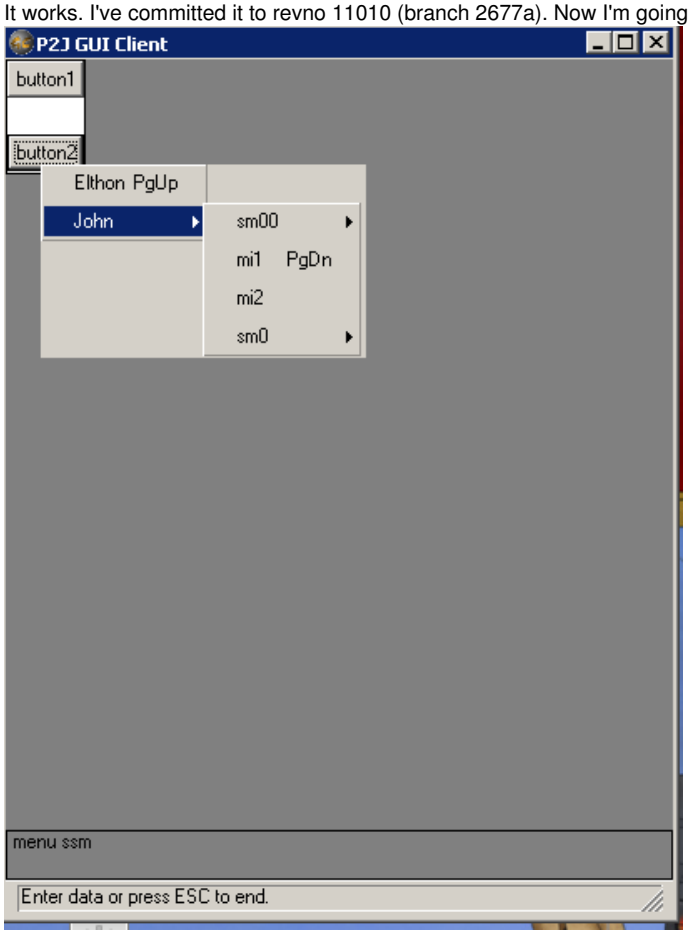
It seems I've solved it. Sorry to trouble. I'll commit it a little later. I'm testing solution at this moment. The reason was related to solution of "blinking" problem. Some logic was moved from mouse* listeners to onFocusGain/onFocusLost listeners to avoid blinking during navigation. And I was forced to

add manual FOCUS_GAINED generation to mouseExit to correctly setup flags. I could manually call source.showBody() but it will leads to blinking, that we wanted to avoid. This solved the body showing issue.

#15 - 11/05/2015 07:43 AM - Vadim Gindin

- File background_extra.png added

It works. I've committed it to revno 11010 (branch 2677a). Now I'm going to fix the following bug:



#16 - 11/05/2015 08:28 AM - Greg Shah

Code Review Task Branch 2677a Revision 11010

The changes look good.

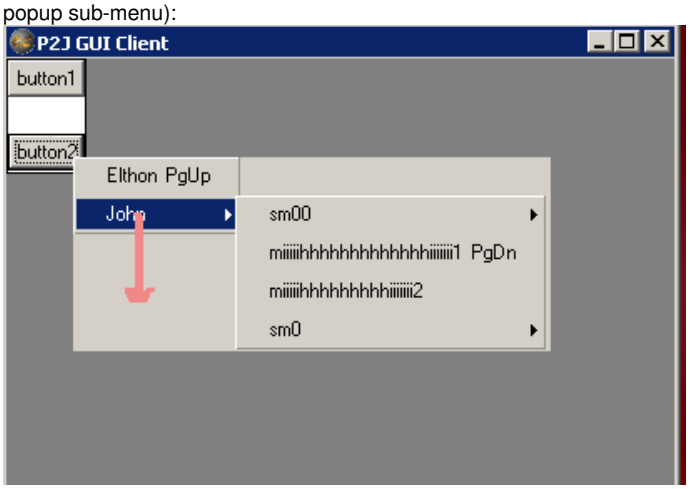
#17 - 11/07/2015 08:41 AM - Vadim Gindin

- File extra_background.png added

I need some advice with the current bug (see the image below).

Procedure: /uast/menus/popup_ext.p.
Bug: extra background rectangle is drawn behind the pop-up menu.

It happen when I move mouse pointer to sub-menu title "John". Its body becomes opened and after that I move mouse pointer down (outside of popup sub-menu):



At this moment additional background rectangle is drawn. At first, I tried to debug dimensions of drawn rectangles in MenuGuiImpl and SubMenuGuiImpl, but I found that none of calls fillRect/fill3DRect/fillPolygon from these classes draw that rectangle. It happen in BorderedPanelGuiImpl.draw() in the inner gd.draw() call for one of clippings in a cycle (lines 152-160):

```
    ..
    List<NativeRectangle> clippings = screen().getClippings(windowId, nrect);
    for (NativeRectangle c : clippings)
    {
        gd.fillRect(c.left() - screenOrigin.x,
                    c.top() - screenOrigin.y,
                    c.width(),
                    c.height());
    }
    ..
```

Actually I'm confused a little. It could be related with the pop-up menu size. Menu sizes rectangle is a rectangle of minimal size, that covers all opened sub-menus. In other words that extra background rectangle conforms to menu size rectangle in the state when sub-menu "John" is opened.

Could you advice me something?

#18 - 11/10/2015 06:09 AM - Vadim Gindin

It seems that all drawing is happen, but after that window is closed without any pauses or messages. Procedure /uast/menu/popup_ext.p other procedure works as usual. How to debug window closing?

#19 - 11/10/2015 08:16 AM - Vadim Gindin

Once again, it is not a configuration/build problem, because simple_sm.p, for example, works as usual. Drawing happen, event processing loop is working, i.e. not finished in ThinClient.waitForWorker(..). BUT: Window is not shown.. By the way Window also has visible=false. Is it OK for default window? If that is the reason, where it must be set? As I'd found, the client side does not support logging configuration. Isn't it?

#20 - 11/10/2015 08:17 AM - Constantin Asofiei

Vadim Gindin wrote:

> Once again, it is not a configuration/build problem, because simple_sm.p, for example, works as usual. Drawing happen, event processing loop is working, i.e. not finished in ThinClient.waitForWorker(..). BUT: Window is not shown.. By the way Window also has visible=false. Is it OK for default window? If that is the reason, where it must be set? As I'd found, the client side does not support logging configuration. Isn't it?

DEFAULT-WINDOW is not-visible by default. Only when some UI statemen targets it (like DISPLAY or MESSAGE), then it is displayed.

I'm looking into your popup_ext.p issue now.

#21 - 11/10/2015 08:32 AM - Constantin Asofiei

Vadim Gindin wrote:

> Once again, it is not a configuration/build problem, because simple_sm.p, for example, works as usual. Drawing happen, event processing loop is working, i.e. not finished in ThinClient.waitForWorker(..). BUT: Window is not shown.. By the way Window also has visible=false. Is it OK for default window? If that is the reason, where it must be set?

Actually, there is a regression related to DEFAULT-WINDOW, introduced by my HIDDEN changes

> As I'd found, the client side does not support logging configuration. Isn't it?

Correct. You can use System.err.println to log messages on client-side.

#22 - 11/10/2015 08:57 AM - Constantin Asofiei

OK, the issue in note 17 is related to repaint calls for the sub-menu. Currently, you are posting a PaintEvent with repaint area equal with the maximum width/height as if all sub-menus are expanded (and this includes space outside of the area actually occupied by a sub-menu). This is not OK. I think is better to post a PaintEvent for each individual expanded sub-menu. I'm thinking something like this in SubMenuGuiImpl.repaint(), so that repaint events are posted only for the rectangles where the sub-menu (and its body) are actually shown:

```
      if (this.isBodyDisplayed())
      {
         repaint(new Rectangle(bodyLocation(), bodyDimension(), screen().coordinates().baseUnits()));

         for (Widget<O> w : widgets())
         {
            if (w instanceof SubMenu)
            {
               w.repaint();
            }
         }
      }
```

#23 - 11/10/2015 01:43 PM - Vadim Gindin

Constantin Asofiei wrote:

> Vadim Gindin wrote:
>
> > Once again, it is not a configuration/build problem, because simple_sm.p, for example, works as usual. Drawing happen, event processing loop is working, i.e. not finished in ThinClient.waitForWorker(..). BUT: Window is not shown.. By the way Window also has visible=false. Is it OK for default window? If that is the reason, where it must be set?
>
> Actually, there is a regression related to DEFAULT-WINDOW, introduced by my HIDDEN changes

Is there some fix already (may be temporary)?

#24 - 11/10/2015 02:19 PM - Constantin Asofiei

Vadim Gindin wrote:

> Constantin Asofiei wrote:
>
> > Vadim Gindin wrote:
> >
> > > Once again, it is not a configuration/build problem, because simple_sm.p, for example, works as usual. Drawing happen, event processing loop is working, i.e. not finished in ThinClient.waitForWorker(..). BUT: Window is not shown.. By the way Window also has visible=false. Is it OK for default window? If that is the reason, where it must be set?
> >
> > Actually, there is a regression related to DEFAULT-WINDOW, introduced by my HIDDEN changes
>
> Is there some fix already (may be temporary)?

Add a DEFAULT-WINDOW:VISIBLE = true in your tests, it will solve the problem for now. I'm working on a fix.

#25 - 11/11/2015 02:43 AM - Vadim Gindin

Constantin Asofiei wrote:

> OK, the issue in note 17 is related to repaint calls for the sub-menu. Currently, you are posting a PaintEvent with repaint area equal with the maximum width/height as if all sub-menus are expanded (and this includes space outside of the area actually occupied by a sub-menu). This is not OK. I think is better to post a PaintEvent for each individual expanded sub-menu. I'm thinking something like this in SubMenuGuiImpl.repaint(), so that repaint events are posted only for the rectangles where the sub-menu (and its body) are actually shown:
> [...]

1. Note that this bug is related for pop-up menu only. When MENUBAR with sub-menus is used, there is not such error.
2. I could be wrong, but I remember that resulting invalidation rectangle is an intersection of rectangles of all child widgets in a hierarchy. So that can broke all repainting. Isn't it? You wrote about something like that in the task #1790 notes 189,409,428. Please have a look.

#26 - 11/11/2015 04:19 AM - Constantin Asofiei

Vadim Gindin wrote:

> Constantin Asofiei wrote:
>
> > OK, the issue in note 17 is related to repaint calls for the sub-menu. Currently, you are posting a PaintEvent with repaint area equal with the maximum width/height as if all sub-menus are expanded (and this includes space outside of the area actually occupied by a sub-menu). This is not OK. I think is better to post a PaintEvent for each individual expanded sub-menu. I'm thinking something like this in SubMenuGuiImpl.repaint(), so that repaint events are posted only for the rectangles where the sub-menu (and its body) are actually shown: [...]
>
> 1. Note that this bug is related for pop-up menu only. When MENUBAR with sub-menus is used, there is not such error.
> 2. I could be wrong, but I remember that resulting invalidation rectangle is an intersection of rectangles of all child widgets in a hierarchy. So that can broke all repainting. Isn't it? You wrote about something like that in the task #1790 notes 189,409,428. Please have a look.

I think this is really an issue of posting a too "aggressive" repaint rectangle. When focus is lost from a sub-menu, a PaintEvent is posted which uses the smallest rectangle as if all sub-menus are expanded. Look how AbstractContainer.getDrawableWidgets works: it goes through all widgets, in z-order, from top to bottom, and stops when all the invalidation rectangles are "consumed". So, as the MENU is higher on z-order than the WindowWorkspace, the MENU's boundaries (which is, again, the smallest rectangle including all sub-menus expanded) will match the only posted rectangle in the ScreenBitmap: so all "invalidation rectangles" are consumed when the MENU is found, and the WindowWorkspace widget never gets repainted - as it is below the MENU.

The idea here is, if a rectangle is posted to ScreenBitmap, is assumed that the entire area needs to be drawn by the widget - ScreenBitmap can not use "transparent" rectangles, as these make no sense. So that's why you get an incorrectly drawn area when focus is lost: the workspace (which is below the MENU) never gets a chance to repaint, as the P2J internal drawing mechanism assumes that if a widget posted a rectangle to be repainted, then that widget is responsible of fully drawing that area, if the widget is the topmost widget.

#27 - 11/11/2015 05:07 AM - Constantin Asofiei

I think we need to modify width() and height() to report only the sub-menus "title" area; after this, override repaint() in sub-menu so that:

1. a PaintEvent is posted with the sub-menu's title
2. a PaintEvent is posted with the sub-menu's body, if it's expanded

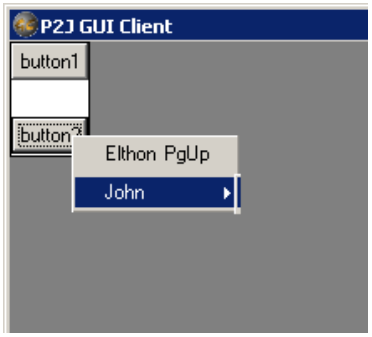
#28 - 11/11/2015 08:40 AM - Vadim Gindin

Constantin, thank you for help! Just one question. How about BorderedPanelGuiImpl? I recall that mentioned erroneous rectangle is drawn from there: BorderedPanelGuiImpl.draw(..) without throwing PaintEvent and using straight call gd.fillRect(...).

P.S. I'm trying to implement your proposals and fixing arising bugs at this moment.

#29 - 11/12/2015 02:09 PM - Vadim Gindin

- File intersection.png added

1. All my attempts to fix the following bug (after I made proposed changes): 

ends with nothing at this moment. I even manually set up repaint rectangle rectangle for sub-menu body but it is not displayed with proposed approach.. Could you advice me how to debug it effectively?
2. OK, the way when the Menu is responsible for drawing it's background is good and WindowWorkspace will not redraw it. But why it is needed to draw rectangles from BorderedPanelGuiImpl.draw() using gd.fillRect()? I.e. workspace unconditionally draws some rectangles..

#30 - 12/03/2015 01:31 PM - Greg Shah

- % Done changed from 0 to 100

- Status changed from New to Closed

#31 - 11/16/2016 12:12 PM - Greg Shah

- Target version changed from Milestone 12 to GUI Support for a Complex ADM2 App

Files

body_drawing.png	3.94 KB	10/27/2015	Vadim Gindin
background_extra.png	8.23 KB	11/05/2015	Vadim Gindin
extra_background.png	4.6 KB	11/07/2015	Vadim Gindin
intersection.png	2.11 KB	11/12/2015	Vadim Gindin