User Interface - Bug #3492

in the web client the mouse focused fill-in field doesn't show the caret

02/27/2018 09:29 AM - Sergey Ivanovskiy

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

Description

History

#1 - 02/27/2018 09:32 AM - Sergey Ivanovskiy

The bug case can be reproduced with ./demo/demo_widgets.p manual GUI test. The cursor isn't displayed if we open the web client and click on the "fill-in2:".

#3 - 02/27/2018 10:01 AM - Greg Shah

- Subject changed from In the web client the mouse focused fill-in field doesn't show the carret. to In the web client the mouse focused fill-in field doesn't show the caret.

#5 - 06/12/2018 10:05 AM - Sergey Ivanovskiy

- Status changed from New to WIP

#6 - 06/12/2018 10:07 AM - Greg Shah

- Start date deleted (02/27/2018)

- Project changed from FWD to User Interface

- Assignee set to Sergey Ivanovskiy

#7 - 06/12/2018 01:30 PM - Sergey Ivanovskiy

This issue can be reproduced for the Swing client too for this simple test program

```
DEF VAR chf AS CHAR  VIEW-AS FILL-IN LABEL "fill-in" FORMAT "x(20)".
DEF VAR chf2 AS CHAR  VIEW-AS FILL-IN LABEL "fill-in2" FORMAT "x(20)".

def frame f2 chf chf2 with side-labels size 60 by 10.

enable all with frame f2.

wait-for close of this-procedure.
```

Also it can be observed that this program is displayed differently by the web client. Two fillins are in two horizontal lines but the Swing client and 4GL displays them correcly in one horizontal line. It is obviously a regression in the trunc.

#8 - 06/12/2018 01:32 PM - Sergey Ivanovskiy

But these differences can be related to the web font setup.

#9 - 06/13/2018 04:16 AM - Sergey Ivanovskiy

It can be observed that the caret is displayed at the moment when the current focus is requested first time for the target field and then it has been overridden by filling rectangle. In the web case the root case is not clear and there are another unknown reasons.

#10 - 06/13/2018 04:01 PM - Sergey Ivanovskiy

It seems that we shouldn't cache a cursor drawing. I temporary tested the web client with the branch 3492a (rev 11265) disabling the graphics cache system and found that the cursor becomes visible if TAB key is used to get the target focus. But mouse clicking doesn't show stable improvements, sometimes the caret becomes invisible if the target widget gets the focus input via mouse clicking. If the graphics cache is enabled, then the caret is erased after the target widget gets the first focus on a TAB/SHIFT+TAB or a mouse click.
Thus for the web client the web cache is important root cause of the caret visibility.
The Swing client looks fine with the branch 3492a (rev 11265).

#11 - 06/13/2018 04:04 PM - Sergey Ivanovskiy

- File demo_widgets_2.p added

I used this simple test with fill-ins widgets.

#12 - 06/14/2018 12:48 PM - Sergey Ivanovskiy

It seems that if we invoke this function twice for the caret drawing, then the second invocation should erase this caret.

```
/**
 * Renders the given image on the canvas at the point (x,y) given by its coordinates in the
 * canvas associated coordinate system.
 *
 * @param    {ImageData} imageData
 *           The imageData to be drawn.
 * @param    {Number} x
 *           X coordinate of the pixel to be drawn.
 * @param    {Number} y
 *           Y coordinate of the pixel to be drawn.
 */
CanvasRenderer.prototype.putImageData = function(imageData, x, y)
{
   if (this.compositeModeOn)
   {
      this.applyCompositeMode(imageData, x, y);
   }
   try
   {
      this.ctx.putImageData(imageData, x, y);
   }
   catch(ex)
   {
      console.log("putImageData(imageData=" + (imageData ? imageData.width : "") + "x" +
                  (imageData ? imageData.height : "") +
                  " data.byteLength=" + (imageData ? imageData.data.byteLength : "") +
                  ", x=" + x + ", y=" + y + ")" + " compositeModeOn=" + this.compositeModeOn);
      console.log(ex.stack);
   }
};

/**
 * Transforms a given image according to the current composite mode. The input image is changed.
 *
 * @param    {ImageData} imageData
 *           The input image to be transformed.
 * @param    {Number} x
 *           X coordinate of the left image corner.
 * @param    {Number} y
 *           Y coordinate of the top image corner.
 */
CanvasRenderer.prototype.applyCompositeMode = function(imageData, x, y)
{
   if (this.compositeModeOn)
   {
      var w = imageData.width;
      var h = imageData.height;
      var screenShot = this.screenCtx.getImageData(x, y, w, h);
      var pixelsInBytes = 4 * w * h;
      for (var i = 0; i < pixelsInBytes; i += 4)
      {
         imageData.data[i]     = imageData.data[i]     ^ screenShot.data[i];
         imageData.data[i + 1] = imageData.data[i + 1] ^ screenShot.data[i + 1];
         imageData.data[i + 2] = imageData.data[i + 2] ^ screenShot.data[i + 2];
         imageData.data[i + 3] = 255;
      }
   }
};
```

#13 - 06/14/2018 04:31 PM - Sergey Ivanovskiy

- File test_canvas_renderer.html added

It is difficult for me to fix XOR implementation because I don't know the requirements. Which drawing effects must this implementation simulate? This test page should be placed under src/com/goldencode/p2j/ui/client and then it can show that if the canvas has a background, then it affects line drawings. The first invocation makes lines visible and the second one hides them. It affects the web cursor visibility too.

#14 - 06/14/2018 04:56 PM - Sergey Ivanovskiy

Ovidiu, could you help. What effect should we expect from xor cursor?

```
   /**
    * Renders a caret for an enabled widget ({@code FillInGuiImpl} or {@code EditorGuiImpl}).
    *
    * @param   gd
    *          The {@code GuiDriver} used for drawing operations.
    * @param   x1
    *          The left extremity of the caret.
    * @param   y1
    *          The top extremity of the caret.
    * @param   x2
    *          The right extremity of the caret.
    * @param   y2
    */
   protected void drawCaret(GuiDriver gd, int x1, int y1, int x2, int y2)
```

```
    {
       gd.setXORComposite();
       setCompositeXorColor(gd);
       gd.drawLine(x1, y1, x2, y2);
       gd.resetComposite();
    }
```

#15 - 06/14/2018 05:02 PM - Sergey Ivanovskiy

There are combinations of background and cursor colors that make lines invisible if we draw a caret.

#16 - 06/14/2018 05:04 PM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> There are combinations of background and cursor colors that make lines invisible if we draw a caret.

You mean the problem is when a single line is being drawn? Can you try something, like changing the caret-drawing from single-line to a 2-3 pixels width rectangle? I mean, I want to make sure the problem is with XOR'ing in web and not something else, as I've noticed missing caret in some cases in Swing, too.

#17 - 06/14/2018 05:07 PM - Sergey Ivanovskiy

I found that all colors in our web are supposed to be opaque. We have no transparent colors like rgba(128,128,128,0.5)!

#18 - 06/14/2018 05:08 PM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Sergey Ivanovskiy wrote:
>
>> There are combinations of background and cursor colors that make lines invisible if we draw a caret.
>
> You mean the problem is when a single line is being drawn? Can you try something, like changing the caret-drawing from single-line to a 2-3 pixels width rectangle? I mean, I want to make sure the problem is with XOR'ing in web and not something else, as I've noticed missing caret in some cases in Swing, too.

I have small fixes for the swing client too.

#19 - 06/14/2018 05:13 PM - Sergey Ivanovskiy

Please review these small fixes for the swing client rev 11266 and 11265 in the branch 3492a.

#20 - 06/14/2018 05:20 PM - Sergey Ivanovskiy

- File deleted (test_canvas_renderer.html)

#21 - 06/14/2018 05:23 PM - Sergey Ivanovskiy

- File test_canvas_renderer.html added

Changed to exaggerate root causes.

#22 - 06/14/2018 05:23 PM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Please review these small fixes for the swing client rev 11266 and 11265 in the branch 3492a.

In FillInGuiImpl.draw, drawCaret must be called only for enabled widgets - if you meant to call it for uninitialized fill-in's, then just add a check if the widget is enabled or not.

#23 - 06/14/2018 05:28 PM - Sergey Ivanovskiy

It seems that drawCaret() checks that this fill-in is enabled.

```
   public void drawCaret()
   {
      // this check is required since the method can be called from multiple drivers
      if (!hasFocus() || !isEnabled())
      {
         return;
      }

      // drawing worker
      drawCaretInt();
   }
```

#24 - 06/14/2018 05:43 PM - Sergey Ivanovskiy

Constantin, if we turn off

```
          <node class="boolean" name="graphicsCached">
```

```
            <node-attribute name="value" value="FALSE"/>
          </node>
```

then with this example demo_widgets_2.p we can observe this effect when pressing and releasing the mouse make the caret visible and then hide it. If the drawing cache is on, then it can draw affected images.

#25 - 06/14/2018 05:50 PM - Sergey Ivanovskiy

Another way is to apply this diff to 3492a and we can observe that the cursor is present on mouse click and it looks working properly.

```
=== modified file 'src/com/goldencode/p2j/ui/client/gui/theme/ClassicTheme.java'
--- src/com/goldencode/p2j/ui/client/gui/theme/ClassicTheme.java    2018-05-03 20:44:34 +0000
+++ src/com/goldencode/p2j/ui/client/gui/theme/ClassicTheme.java    2018-06-14 21:47:28 +0000
@@ -2393,10 +2393,10 @@
     */
    protected void drawCaret(GuiDriver gd, int x1, int y1, int x2, int y2)
    {
-      gd.setXORComposite();
+//      gd.setXORComposite();
       setCompositeXorColor(gd);
       gd.drawLine(x1, y1, x2, y2);
-      gd.resetComposite();
+//      gd.resetComposite();
    }

    /**
```

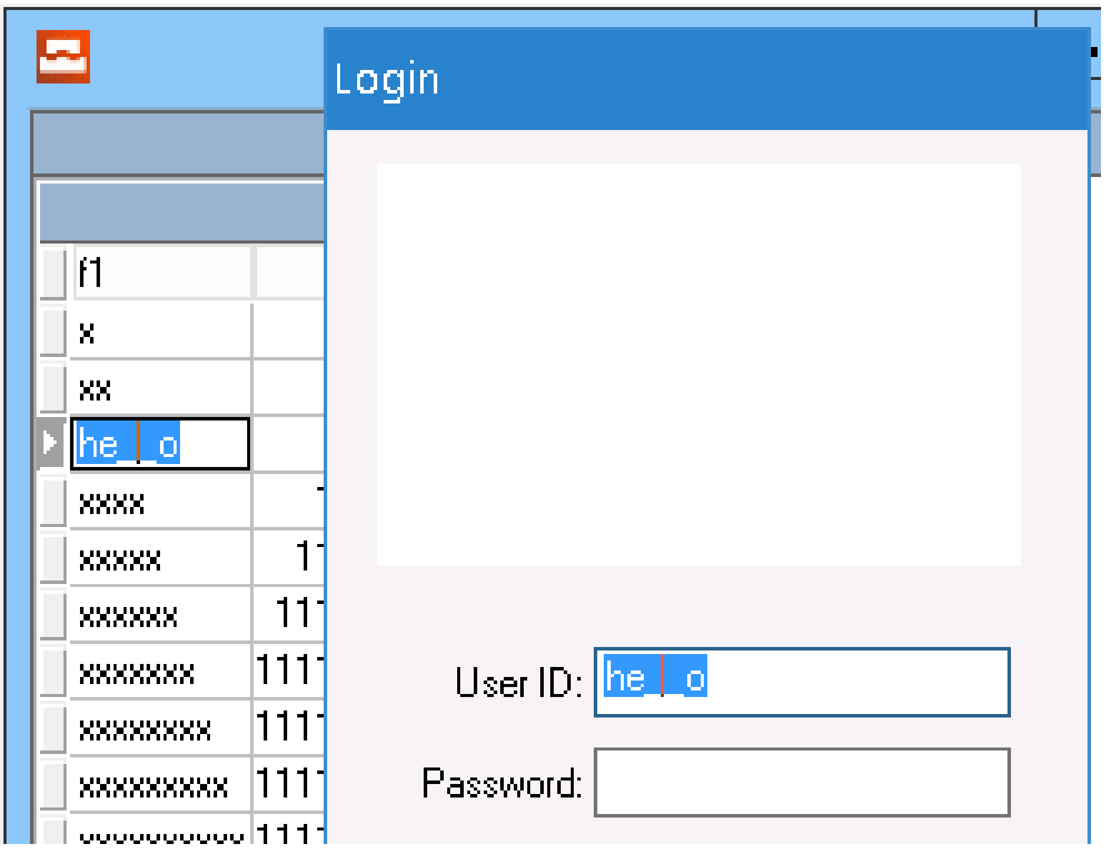
#26 - 06/15/2018 04:20 AM - Ovidiu Maxiniuc

- File Screenshot_2018-06-15_11-10-47.png added

Sergey Ivanovskiy wrote:

> Ovidiu, could you help. What effect should we expect from xor cursor?
> [...]

The problem is not while drawing on white background. In this case, the effect is the same (XOR-ing the white will give black, same as drawing a black line on top). The special handling is necessary when the background of the caret is coloured. Please see the screenshot:



The left in a testcase of a browse cell edited in ABL. The cursor line is XOR-ed the blue selection background (resulting some kind of red-brown) and the white line of underscore (making the overlapped pixel black).
In the right there is a login screen running on FWD, with a user id also selected. The very same colour effects can be seen.

The advantage of the XOR operation is that if drawn twice with same parameter, the first line is reversed. I think this is why ABL uses it. As you see, the effect is the same in ABL and FWD. I am unable to tell why is the small difference in colour, but it might be caused because the ABL runs on Windows8 while FWD uses a Windows10 theme.

#27 - 06/15/2018 06:34 AM - Sergey Ivanovskiy

Thank you. I see that the web xor transformation is correct but if these transformations are repeated, then the resulted image becomes highlighted and it can affect recorded cached images. Thus, I don't see now the good solution to cursor visibility issue. There is only one to prevent drawing caching by
this setting

```
<node class="boolean" name="graphicsCached">
  <node-attribute name="value" value="FALSE"/>
</node>
```

#28 - 06/15/2018 06:43 AM - Greg Shah

> I see that the web xor transformation is correct but if these transformations are repeated, then the resulted image becomes highlighted and it can affect recorded cached images.

I don't understand the root cause. Is the operation repeated when it should not be repeated?

#29 - 06/15/2018 06:53 AM - Sergey Ivanovskiy

Greg Shah wrote:

> > I see that the web xor transformation is correct but if these transformations are repeated, then the resulted image becomes highlighted and it can affect recorded cached images.
>
> I don't understand the root cause. Is the operation repeated when it should not be repeated?

If you just click on the fill-in, then the cursor should be repainted.

#30 - 06/15/2018 06:56 AM - Greg Shah

> If you just click on the fill-in, then the cursor should be repainted.

I don't understand how this is the root cause. This sounds like a description of the problem that occurs, not the cause of the problem.

#31 - 06/15/2018 07:00 AM - Sergey Ivanovskiy

Greg Shah wrote:

> > If you just click on the fill-in, then the cursor should be repainted.
>
> I don't understand how this is the root cause. This sounds like a description of the problem that occurs, not the cause of the problem.

I mean that if the cursor will be painted over the current screen shot, then it becomes invisible. I tested the current version of 3492a. You can check this evidence by setting

```
<node class="boolean" name="graphicsCached">
  <node-attribute name="value" value="FALSE"/>
</node>
```

#32 - 06/15/2018 07:03 AM - Greg Shah

Are we caching something (the cursor) that should not be cached?

OR

Are we executing a primitive operation (xor for the cursor) a second time when we should not execute it?

#33 - 06/15/2018 07:24 AM - Sergey Ivanovskiy

I don't know how to fix these issues. The cursor is drawn if draw() has been invoked. If the widget background is repainted, then the cursor should be repainted over this background. But in the web case we record drawings if the target clipping region is contained inside the current clipping regions and then their cached versions can be used by the web client.

#34 - 06/15/2018 07:34 AM - Sergey Ivanovskiy

I mean that if we cache drawings that repaints cursor even one time or twice or more, then the result image can have an invisible cursor.

#35 - 06/15/2018 09:13 AM - Greg Shah

We could make a driver-level primitive op for drawing the cursor. Then each driver would be able to handle it differently.

#36 - 06/15/2018 10:45 AM - Constantin Asofiei

I don't think the problem is with graphics cached - I think the problem is with excessive drawing of the caret (and the widget itself) - as we use XOR, if we draw it an even number of times, it will be cleared; if drawn an odd number of times, it will be shown.

So, I'm trying to cut down on the drawing(s) and stabilize it.

#37 - 06/15/2018 12:41 PM - Constantin Asofiei

Sergey, please test 3492a rev 11267 - for Hotel GUI, it solves issues in Web. I know the clicking in the FILL-IN doesn't position the caret in 3492a, but this is fixed in 3507a.

Greg: please review.

#38 - 06/15/2018 02:32 PM - Greg Shah

Code Review Task Branch 3492a Revision 11267

1. It is not clear to me how safe it is to bypass flushing on entry to the TC.independentEventDrawingBracket(). What is the thinking behind this?

2. This is a comment from the changes:

> if this is called from outside of the actual driver drawing mechanism, ignore it

The association of "actual driver drawing mechanism" with invalidate == 0 is hard for me to see. Can you help me understand why this is a safe assumption?

3. The ThinClient.independentEventDrawingBracket() javadoc is not accurate with the current changes.

#39 - 06/15/2018 02:41 PM - Constantin Asofiei

Greg Shah wrote:

> Code Review Task Branch 3492a Revision 11267
>
> 1. It is not clear to me how safe it is to bypass flushing on entry to the TC.independentEventDrawingBracket(). What is the thinking behind this?

When independentEventDrawingBracket is being called from another eventDrawingBracket call (i.e. nested calls), we currently flush all drawing - see

OutputManager.resetInvalidate. But these drawings don't belong to this independent bracket - they belong to the root eventDrawingBracket call. The idea here is to make this really independent - as these are saved:

1. the previous invalidate depth
2. the previous ScreenBitmap instance

and restore these once the 'independent drawing' work is done - the root eventDrawingBracket call will have a chance to do its drawing, and not be 'stolen' by an independentEventDrawingBracket call.

> 2. This is a comment from the changes:
>
> > if this is called from outside of the actual driver drawing mechanism, ignore it
>
> The association of "actual driver drawing mechanism" with invalidate == 0 is hard for me to see. Can you help me understand why this is a safe assumption?

There are cases where caret drawing is being called from i.e. event processing loop - this is not OK. Caret (and actually all) drawing must be done when we reach the root eventDrawingBracket call, and when this call is being processed, the invalidate will become zero. So that's why the test checks for isInvalidate() - this means the drawing is being performed by the root, and final drawing bracket.

> 3. The ThinClient.independentEventDrawingBracket() javadoc is not accurate with the current changes.

Will fix it.

#40 - 06/15/2018 02:45 PM - Sergey Ivanovskiy

I checked Hotel_GUI and didn't find obviously seen visual bugs.

#41 - 06/15/2018 02:51 PM - Greg Shah

Thank you for the details. I'm OK with the code changes.

Is there anything we are waiting for before this branch can go into trunk?

Sergey: Please test the customer's large GUI application using this branch. Also, please start ChUI regression testing.

#42 - 06/15/2018 03:41 PM - Sergey Ivanovskiy

OK, the customer application has no obvious visual bugs with 3492a. Started CHUI testing.

#43 - 06/16/2018 04:02 AM - Sergey Ivanovskiy

The CHUI regression tests were all failed due to hanging, I checked the manual start, but run_regression.sh server-startup has been hung until now. I don't know it is due to the last changes in 3492a or it is a devsrv01 issue.

#44 - 06/19/2018 01:10 PM - Constantin Asofiei

Sergey, see 3492a rev 11269 for a fix. Please restart MAJCI ChUI regression testing and check Hotel GUI and the customer app for any regressions.

#45 - 06/19/2018 02:01 PM - Greg Shah

Code Review Task Branch 3492a Revision 11269

I'm good with the changes.

#46 - 06/20/2018 08:49 AM - Constantin Asofiei

3492a rev 11270 fixes balancing issues with drawing invalidation brackets - exposed by the change in TC.independentEventDrawingBracket.

#47 - 06/20/2018 03:47 PM - Constantin Asofiei

Constantin Asofiei wrote:

> 3492a rev 11270 fixes balancing issues with drawing invalidation brackets - exposed by the change in TC.independentEventDrawingBracket.

Looks like some cases where I tried to balance the drawing's invalidate were not needed, they were being 'balanced' via tk.resetInvalidate(...) called from a ConditionException catch block.

This only case is related to TC.clearWorker - fixed in 3492a rev 11271.

#48 - 06/20/2018 09:36 PM - Sergey Ivanovskiy

Planning to run regression tests.

#49 - 06/21/2018 12:43 PM - Sergey Ivanovskiy

The last regression tests run is hanging. Planning to rebase now and to run tests again.

#50 - 06/21/2018 02:53 PM - Sergey Ivanovskiy

Rebased and the current revision 11272 (3492a) is under chui testing now.

#51 - 06/22/2018 01:11 AM - Sergey Ivanovskiy

- File alert-msg-1.p added

Constantin, please try this program alert-msg-1.p with 3492a. I don't understand it is a real issue or it is due to new changes in application directory, as

```
        <node class="strings" name="resource-jars">
          <node-attribute name="values" value="testcases"/>
        </node>
```

#52 - 06/22/2018 01:12 AM - Sergey Ivanovskiy

The last chui regression tests were failed.

#53 - 06/26/2018 01:22 PM - Sergey Ivanovskiy

Planning to rebase and to restart CHUI testing in order to eliminate failed cases.

#54 - 06/27/2018 03:18 PM - Sergey Ivanovskiy

The current rev 11273 (over trunc rev 11267) has been passed main regression tests. Planning to rebase 3492a now.

#55 - 06/27/2018 04:51 PM - Sergey Ivanovskiy

Now 3492a has the current rev 11274 over the trunc rev. 11268. It seems that Hotel Gui has no visible artifacts with this revision. Should this branch be merged into the trunc?

#56 - 06/27/2018 05:42 PM - Eric Faulhaber

Sergey Ivanovskiy wrote:

> Now 3492a has the current rev 11274 over the trunc rev. 11268. It seems that Hotel Gui has no visible artifacts with this revision. Should this branch be merged into the trunc?

Yes, as Constantin already requested via email.

#57 - 06/27/2018 11:59 PM - Sergey Ivanovskiy

Planning to merge into the trunc now.

#58 - 06/28/2018 12:41 AM - Sergey Ivanovskiy

3492a has been merged/committed into the trunk as a bzr revision 11269 and archived.

#59 - 06/28/2018 07:12 AM - Greg Shah

- Status changed from WIP to Closed

- Subject changed from In the web client the mouse focused fill-in field doesn't show the caret. to in the web client the mouse focused fill-in field doesn't show the caret

- % Done changed from 0 to 100

Files

demo_widgets_2.p	449 Bytes	06/13/2018	Sergey Ivanovskiy
test_canvas_renderer.html	6.01 KB	06/14/2018	Sergey Ivanovskiy
Screenshot_2018-06-15_11-10-47.png	4.97 KB	06/15/2018	Ovidiu Maxiniuc
alert-msg-1.p	279 Bytes	06/22/2018	Sergey Ivanovskiy