User Interface - Bug #2887

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

web client flashing/drawing artifacts

11/27/2015 05:09 PM - Greg Shah

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

Description

Related issues:

Related to User Interface - Support #2672: GUI web client performance improve...	**New**
Related to User Interface - Feature #3246: reduce the amount of data being se...	**Closed**

History

#1 - 11/27/2015 06:20 PM - Sergey Ivanovskiy

The issue is to smooth flashing effects. The proposed solution creates the off screen canvas and the window canvas. CanvasRenderer draws on the off screen canvas, but at the end of the drawing circle the off screen image is blasted on the window canvas.
On 11/27/2015 05:59 PM, Greg Shah wrote:

> I think this makes good sense. We may also need to change the cursor (to the hourglass) while this is happening, so that the user gets some feedback that we are "working".
>
> Please implement the changes on your local system and get them tested with both the customer's GUI application and the standalone testcases. If it doesn't cause any regressions, then we will get it reviewed and into 2677b.

#2 - 11/27/2015 06:51 PM - Sergey Ivanovskiy

- File offscreenBuffer_4.txt added

Greg, please review this diff. It passes 454 testcase for the customer. The changes are to assign p2j.socket.js a function to post user's messages to JS UI thread. These posted messages must be executed sequentially by the browser UI thread. Thus the browser becomes more responsive between these tasks. If a drawing message arrives from the web socket, then it changes the cursor style to be a wait one and posts a drawing task in UI thread, and next posts a restore cursor task in UI thread.

#3 - 11/27/2015 06:57 PM - Sergey Ivanovskiy

- File deleted (offscreenBuffer_4.txt)

#4 - 11/27/2015 06:59 PM - Sergey Ivanovskiy

- File offscreenBuffer_4.txt added

it fixes the wait cursor issue. Also it passes simple tests.

#5 - 12/10/2015 09:29 AM - Greg Shah

I'm adding some of the early discussion here to record it for future reference:

> On 11/27/2015 01:49 PM, Serg Ivanovskiy wrote:
>
> > Greg,
> >
> > I guess that the tab drawings are split to the several drawing circles. Selecting new tab shows several steps: the panel appears, than elements. But the visual effect doesn't satisfy me.
> > Now, I am working on the waiting cursor. For this I need call drawing primitives inside JS event bracket.
> > setTimeout(
> > function() {
> > // some drawing operation
> > }, 1);
> > It should help to do JS client more responsive, because the all single draw() function is splitted on the sequential JS events.
> >
> > Sergey
> >
> > On 11/27/2015 09:08 PM, Greg Shah wrote:
> >
> > > Sergey,
> > >
> > > > but it hasn't improved the performance yet
> > >
> > > I wasn't expecting it to perform better. I was just hoping it would reduce the flashing/redrawing. Does it have any positive effect in this way?
> > >
> > > Thanks,
> > > Greg
> > >
> > > On 11/27/2015 11:55 AM, Serg Ivanovskiy wrote:
> > >
> > > > Greg,
> > > >
> > > > I am testing now, but it hasn't improved the performance yet. We have two known bottlenecks:
> > > > 1) The pixels operations
> > > > 2) The text size requests from the java web client to JS client.
> > > >
> > > > Sergey
> > > >
> > > > On 11/27/2015 05:59 PM, Greg Shah wrote:
> > > >
> > > > > Sergey,
> > > > >
> > > > > I think this makes good sense. We may also need to change the cursor (to the hourglass) while this is happening, so that the user gets some feedback that we are "working".
> > > > >
> > > > > Please implement the changes on your local system and get them tested with both the customer's GUI application and the standalone testcases. If it doesn't cause any regressions, then we will get it reviewed and into 2677b.
> > > > >
> > > > > Thanks,
> > > > > Greg
> > > > >
> > > > > On 11/26/2015 02:11 PM, Serg Ivanovskiy wrote:
> > > > >
> > > > > > Greg, Constantin,
> > > > > >
> > > > > > Today I have experimented with the off screen buffer for the web client and found that it can help to eliminate the drawing artifacts like in the attachment screen.
> > > > > > The method creates the off screen canvas and the actual canvas and CanvasRenderer draws on the off screen canvas, but at the end of the drawing circle the off screen image is blasted on to the actual canvas. The only the mouse resize handler needs to be changed a little but the p2j.screen.js is changed by substituting the drawing on the off screen canvas. What do you think it make sense to apply for 2677b?
> > > > > >
> > > > > > Now I have run the customer app conversion.

Sergey

#6 - 12/10/2015 09:31 AM - Greg Shah

Then there was this set of responses:

On 11/27/2015 05:10 PM, Greg Shah wrote:

> Sergey,
>
> Please post the diff and a description of the approach to #2887.
>
> Thanks,
> Greg
> On 11/27/2015 04:54 PM, Serg Ivanovskiy wrote:
>
> > Greg,
> >
> > I think we should do this step to move to the offscreen buffer it works properly and only the waiting cursor needs to change the logic to do JS client (p2j.socket.js) to deliver messages in the sequence to UI browser thread, these messages have users type. Thus I think p2j.socket.js should be like messages deliver. But now if we execute only drawRectangles message in JS UI bracket, then we have a problem with the message sequences. Thus, users UI messages wait to be performed and we have another type of messages from the web socket and these two types can be messed if we execute only one drawRectangles message in JS UI bracket.
> > I would like to commit my diff. I don't know what the task relates to this work.
> >
> > Sergey
> >
> > On 11/28/2015 12:37 AM, Greg Shah wrote:
> >
> > > We will start the optimization work on the web client soon. For now, if we can't get the flashing to be reduced, then we can postpone further work on this.
> > >
> > > Font/text metrics optimization:
> > >
> > > 1. Share metrics across clients.
> > > 2. Save/restore font/text metrics across server restarts.
> > >
> > > The question is: how "sharable" are those results? Such an approach would eliminate font/text metrics calls to the client for many common paths.
> > >
> > > Greg
> > > On 11/27/2015 03:50 PM, Constantin Asofiei wrote:
> > >
> > > > Sergey,
> > > >
> > > > You are correct, the tabs are simulated using many individual UI statements (DISPLAY, HIDE, VIEW, etc). Each one of these statements is atomic - it does the drawing in its own batch. This results in lots of "chatter" between the Web client and its associated java client...
> > > >
> > > > A next improvement would be to determine if there are any duplicated draw batches, for the same UI statement call: if there are, and we eliminate these, there will be some improvement.
> > > >
> > > > Another check would be to debug the 4GL code and see if we are not missing any kind of "buffering" optimization for drawing (i.e. maybe some missing/overlooked case in #2809, where frame is made hidden, widget drawn, and after that made visible back).
> > > >
> > > > Beside these, we should use some profiler to determine where the JS code can be optimized/improved.

Thanks,
Constantin

#7 - 12/10/2015 09:33 AM - Greg Shah

And there was this discussion of the wait timer:

On 11/27/2015 05:09 PM, Serg Ivanovskiy wrote:

> Greg,
>
> I would like to commit this version at first. And only if it satisfies all tests because I performed only small part of them, then it can be done more complicated, like we discussed. In parallel planning to improve line drawings by pixels using the previous method to put a pixel in the off screen data array at the correct position and then use putImageData to write to the offscreen buffer. It should work. Then compare results.
>
> Sergey
>
>> -------- Forwarded Message --------
>> Subject: Wait cursor Re: Experiments with offscreenBuffer
>> Date: Fri, 27 Nov 2015 22:31:11 +0300
>> From: Serg Ivanovskiy <sbi@goldencode.com>
>> Reply-To: sbi@goldencode.com
>> To: ges@goldencode.com, Constantin Asofiei <ca@goldencode.com>
>>
>> Greg,
>>
>> I think if we change the cursor style inside the one single event, the browser can't display it due to it is busy.
>> May be it is not correct. Please look at http://stackoverflow.com/questions/192900/wait-cursor-over-entire-html-page.
>>
>> Sergey
>>
>> On 11/27/2015 10:22 PM, Greg Shah wrote:
>>
>>> Sergey,
>>>
>>> Why does the cursor have to be set using a timer? Can't we change it at the beginning of the drawing cycle and change it back at the end?
>>>
>>> Greg

#8 - 12/10/2015 10:21 AM - Greg Shah

Code Review offscreenBuffer_4.txt

1. Is it safe to assign document.body.className directly? It will override all styles that are brought in by the previous value. We only want to override the cursor-related value. I guess just adding and removing this class would be better.

2. I would rather NOT have the hardcoded dependency on the DOJO "claro" class name (p2j.defaultBodyHtmlClass = "claro";). Shouldn't we just save the current default value using p2j.defaultBodyHtmlClass = document.body.className;?

3. I see this.canvas.style.visibility = "hidden"; but I don't see where this gets changed to allow drawing to be seen. I must be missing (or misunderstanding) something.

4. My primary concern with the setTimeout() additions to p2j.socket.js is that it seems fragile. Because it is implemented separately for every message, it increases the nesting level and the difficulty of getting the closures correct. This function was already too long and poorly factored and this increases the size and complexity. I think we probably need to move the message-specific processing into its own function to make the switch() condensed and readable. I'd like to see the setTimeout() done in a generic way instead of custom for each message. Perhaps it makes sense to process the entire switch() inside a setTimeout()? Let's discuss this further.

#9 - 12/10/2015 10:23 AM - Greg Shah

I see my mistake about the "hidden" issue. We toggle visibility in Window.prototype.setVisible() using this.canvas.style.visibility = visible ? "visible" : "hidden";. Ignore that question.

#10 - 12/10/2015 11:32 AM - Sergey Ivanovskiy

> Code Review offscreenBuffer_4.txt
>
> 1. Is it safe to assign document.body.className directly? It will override all styles that are brought in by the previous value. We only want to override the cursor-related value. I guess just adding and removing this class would be better.
>
> 2. I would rather NOT have the hardcoded dependency on the DOJO "claro" class name (p2j.defaultBodyHtmlClass = "claro";). Shouldn't we just save the current default value using p2j.defaultBodyHtmlClass = document.body.className;?

I agree we can save the current body class name and then change the css properties by adding new css class document.body.className += (" " + p2j.waitCursorClass);). The reason to use p2j.defaultBodyHtmlClass = "claro"; is that this template page, index.html, sets directly the "claro" css class for the "body" element.

> 4. My primary concern with the setTimeout() additions to p2j.socket.js is that it seems fragile. Because it is implemented separately for every message, it increases the nesting level and the difficulty of getting the closures correct. This function was already too long and poorly factored and this increases the size and complexity. I think we probably need to move the message-specific processing into its own function to make the switch() condensed and readable. I'd like to see the setTimeout() done in a generic way instead of custom for each message. Perhaps it makes sense to process the entire switch() inside a setTimeout()? Let's discuss this further.

Planning to check if it is possible to use setTimeout() on the outer code level since we need to change the current cursor to have the wait style if the drawing is in process and to preserve the messages order, thus we can't have nested user's events created by setTimeout.

#11 - 12/10/2015 01:33 PM - Sergey Ivanovskiy

- File offscreenBuffer_5.txt added

Greg, this code change is improved according your review. I can commit the changes to 1811t if you approve it.

#12 - 12/17/2015 09:56 AM - Greg Shah

Code Review offscreenBuffer_5.txt

This version is better.

Constantin: please review this and provide feedback.

I have some code formatting thoughts:

1. When there is code like this:

```
               setTimeout(
                     function()
                     {
                        messageHandler(message);
                     }, 1);
```

I think it is better if we do this:

```
               setTimeout(function() { messageHandler(message); }, 1);
```

As long as it fits within our line-length limit, this reads better, in my opinion.

2. Code like this:

```
         // resize the offscreen canvas
         win.resize(mThis.resizeArea.width, mThis.resizeArea.height);
         // set dimension for the window canvas
         win.canvas.width  = mThis.resizeArea.width;
         win.canvas.height = mThis.resizeArea.height;
         // blast the offscreen image to the window canvas
         win.ctx.drawImage(win.offscreenCanvas, 0, 0);
```

Is better presented like this:

```
         // resize the offscreen canvas
         win.resize(mThis.resizeArea.width, mThis.resizeArea.height);

         // set dimension for the window canvas
         win.canvas.width  = mThis.resizeArea.width;
         win.canvas.height = mThis.resizeArea.height;

         // blast the offscreen image to the window canvas
         win.ctx.drawImage(win.offscreenCanvas, 0, 0);
```

The idea: it is hard to read the comments and see the logical grouping of the statements that match the comments without the extra blank lines.

3. When there is code like this:

```
      /** the body html css class name to display wait cursor */
      p2j.waitCursorClass = " wait";
      /** the body html css class name to display default cursor */
      p2j.bodyHtmlClass = document.body.className;
      /** to display the wait cursor for the document */
      p2j.displayWaitCursor = function ()
```

Our coding standards require a blank line in between these definitions:

```
      /** the body html css class name to display wait cursor */
      p2j.waitCursorClass = " wait";

      /** the body html css class name to display default cursor */
      p2j.bodyHtmlClass = document.body.className;

      /** to display the wait cursor for the document */
      p2j.displayWaitCursor = function ()
```

I think this is needed, even inside a closure in JS to enhance readability.

Here is another example:

```
   /** The stack capacity, the last visible history.*/
   var isOn;
   /**
    * Initialize the logger with the storage capacity.
    *
    * @param    {Number}
    *           The log stack capacity.
    * @param    {boolean}
    *           The boolean value that indicates the log is on/off.
    */
   me.init = function(capacity, on)
```

It should be like this:

```
   /** The stack capacity, the last visible history.*/
   var isOn;

   /**
    * Initialize the logger with the storage capacity.
    *
    * @param    {Number}
    *           The log stack capacity.
    * @param    {boolean}
    *           The boolean value that indicates the log is on/off.
    */
   me.init = function(capacity, on)
```

I've noticed that you sometimes remove single blank lines that are already there. It is OK to remove extra blank lines, but usually if there is just a single blank line, it is there for a reason.

#13 - 12/17/2015 10:10 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Greg, this code change is improved according your review. I can commit the changes to 1811t if you approve it.

Sergey, the code looks OK; only one note: on p2j.js:100, is the " wait" (with the space prefix) in p2j.waitCursorClass = " wait"; really correct?

#14 - 12/17/2015 10:18 AM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Sergey, the code looks OK; only one note: on p2j.js:100, is the " wait" (with the space prefix) in p2j.waitCursorClass = " wait"; really correct?

I added a space to the wait css class because it is used to change the cursor style for the document:

```
+          if (document.body.className === p2j.bodyHtmlClass)
+          {
+             document.body.className += p2j.waitCursorClass;
+          }
```

#15 - 12/17/2015 03:05 PM - Sergey Ivanovskiy

Greg, please review, the committed revision 10967 has the target fixes.

#16 - 12/21/2015 04:35 PM - Greg Shah

Code Review Task Branch 1811t Revision 10967

Although I have been wanting to remove the DOJO dependency from our GUI web client, the class management helpers in that library may be worth using. Our current approach of editing the text and then copying back a statically saved version of the style is fragile. Changes elsewhere in the client that affect the style property can break this code easily. The DOJO helpers on the other hand can make adding and removing a class both simple and must safer. Please look into those and see if my guess is correct that they can help us.

Otherwise I think the changes are good.

#17 - 01/14/2016 02:29 PM - Greg Shah

- % Done changed from 0 to 100

- Status changed from New to Closed

All requested changes are present in 1811t as of rev 10971.

#18 - 11/16/2016 12:12 PM - Greg Shah

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

#19 - 01/03/2018 02:05 PM - Greg Shah

- Related to Feature #3246: reduce the amount of data being sent to the client-side when an UI attribute is being changed added

Files

offscreenBuffer_4.txt	67.8 KB	11/27/2015	Sergey Ivanovskiy
offscreenBuffer_5.txt	60.6 KB	12/10/2015	Sergey Ivanovskiy