User Interface - Feature #2606

Feature # 2252 (Closed): implement GUI client support

implement GUI runtime support for SELECTION-LIST

07/10/2015 10:16 AM - Greg Shah

Status:	Closed	**Start date:**	07/10/2015
Priority:	Normal	**Due date:**	
Assignee:	Eugenie Lyzenko	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	GUI Support for a Complex ADM2 App		
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 07/10/2015 10:19 AM - Greg Shah

- Parent task set to #2252

This task is for full implementation of the GUI widget implementation for SELECTION-LIST. Both static and dynamic usage is needed.

As far as I know, the attributes and methods which we need are already implemented, except for MAX-CHARS. However, if you see some important attributes and methods which you believe are needed, please note them here and we will discuss.

#2 - 08/17/2015 11:19 AM - Greg Shah

Please add SCROLL-TO-ITEM() method support as part of this task.

#3 - 08/18/2015 08:55 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10925.

This is just placeholder for selection-list implementation. The question I'm thinking now is if we need to keep ChUI implemented approach in GUI adding another entity for real implementation inside the SelectionListGuiImpl - SelectionListBodyGuiImpl. Or we can use simpler approach combining two widget into single one to have better implementation.

#4 - 08/19/2015 10:27 AM - Eugenie Lyzenko

Rebased task branch 2606a from P2J trunk revision 10926, new branch revision is 10927.

#5 - 08/20/2015 11:12 AM - Eugenie Lyzenko

Rebased task branch 2606a from P2J trunk revision 10927, new branch revision is 10928.

#6 - 08/20/2015 09:07 PM - Eugenie Lyzenko

- File sel_list_p2j_test0_0820.jpg added

Task branch 2606a for review updated to revision 10929.

This is very base selection-list implementation for single selection mode as starting point. The widget accepts keys and changes the selection accordingly. The screen shot for results is attached.

The approach taken is to preserve existed abstraction logic for selection-list separating common, ChUI and GUI code in respective classes to isolate

UI specifics.

#7 - 08/21/2015 07:24 PM - Eugenie Lyzenko

- File sel_list_p2j_test1_0821.jpg added

Task branch 2606a for review updated to revision 10930.

This update adds support for multiple selection mode in GUI. Also fixed the incorrect getting of the screen value in SELECTION-LIST. It can exceed the format value in multiple selection mode. The sample for multiple selection mode is also attached. Mouse is not yet supported.

The next step will be mouse handling for both modes and scrollbars support.

#8 - 08/23/2015 04:51 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10931.

This adds the mouse support for GUI selection-list is base functions.

#9 - 08/23/2015 05:11 PM - Eugenie Lyzenko

Rebased task branch 2606a from P2J trunk revision 10928, new branch revision is 10932.

#10 - 08/24/2015 09:53 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10933.

This adds the generic scrollbar support for GUI selection-list. The vertical scrollbar is fully working while horizontal one is under construction. The investigation shows the 4GL scroll horizontally per character base. Meaning one scroll shifts the visible are one char to the left/right. Implementing is in progress.

#11 - 08/25/2015 07:47 AM - Greg Shah

Do you have things to work on that do not involve scrolling? Hynek's 2424c update which is close to finishing testing, has a very new scrolling approach. It cleans up many problems and rationalizes the classes used. It would be better to start with that version instead of building things from scratch. Please work on other things first.

#12 - 08/25/2015 07:56 AM - Eugenie Lyzenko

Greg Shah wrote:

> Do you have things to work on that do not involve scrolling? Hynek's 2424c update which is close to finishing testing, has a very new scrolling approach. It cleans up many problems and rationalizes the classes used. It would be better to start with that version instead of building things from scratch. Please work on other things first.

OK.

#13 - 08/25/2015 08:45 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10934.

This update contains suspended implementation scrolling for horizontal direction. As we discussed I have frozen this work. Instead the support for SCROLL-TO-ITEM() method has been started to implement. Currently the conversion is ready. Starting to implement runtime part(both server and client sides).

#14 - 08/26/2015 08:56 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10935.

This update completes the support for SCROLL-TO-ITEM() method by adding the runtime implementation. The idea is to minimize server-client relationship by implementing everything via single call with int index as item to scroll. Note the server side works with 1-based index like 4GL while the client side works with 0-based index. This transformation is performing when LogicalTerminal calls ThinClient for service. The reason is to be compatible with existed P2J code for ChUI mode where the index is 0-based(and actually it is more machine-natural for int value).

The next step is to implement dynamic selection-list where we also can check how the SCROLL-TO-ITEM() method will work for unwrapped handle case.

New GUI issue has been found during testing. If we have pause() statement when event loop is active(for example wait for some button to be pressed) - we need to press spacebar to resume after pause() processing. And after pressing spacebar the pause dismissing but the SPACEBAR key is still routing to the current widget under focus. While in 4GL it is not the case and pause() eats the unblock key.

#15 - 08/27/2015 05:02 PM - Eugenie Lyzenko

Several testing shows the selection-list widget is working properly in case of the dynamically creation. Also the implemented SCROLL-TO-ITEM() method works also fine with dynamic selection-list.

So the remaining issues with selection-list is pending horizontal/vertical scrolling and size scaling issue similar one we had with combo-box plus some general testing to identify additional problematic places. Working with scaling issue first.

#16 - 08/27/2015 09:23 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10936.

This update for review has fix for incorrect widget width getting on server side. Also it implements scaling approach for selection-list. Base tests shows same width in both pixels and characters in 4GL and P2J. The issue found is not properly auto location and painting for vertical scrollbar. Need to adjust.

Things to do is to make additional testing for different size/font options to be properly scaled. Also to check proper vertical dimension computing for single row and whole widget.

Confirm the dynamic version of the selection list is working fine.

#17 - 08/28/2015 01:11 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10937.

This improves height scaling. Preparing to rebase.

#18 - 08/28/2015 04:14 PM - Eugenie Lyzenko

Rebased task branch 2606a from P2J trunk revision 10929, new branch revision is 10938.

#19 - 08/28/2015 10:20 PM - Eugenie Lyzenko

- File sel_list_p2j_test5_3_0828.jpg added

- File sel_list_test5_3_4gl_0.jpg added

- File sel_list_p2j_test5_0828.jpg added

- File sel_list_test5_4gl_1.jpg added

The new code samples to compare. As we can see the scrollbar does not have the painting for disabled state. But the widget scaling is working fine.

#20 - 08/28/2015 11:12 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10939.

This update fixes the scrollbar positioning. However I've found the scrollbar does not have the drawing code for disabled state is required in selection-list. This is the update that reflects the pictures uploaded in previous note.

#21 - 08/29/2015 08:28 AM - Eugenie Lyzenko

Another TODO work is to verify SIZE PHRASE working and tooltip functionality.

#22 - 08/31/2015 12:28 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10940.

This update completes handling of the SIZE PHRASE option for selection-list widget(including ability to specify explicit dimensions in pixels). The rule frame_generator.xml has been modified to be able to handle size-pixels options for widgets that are coming from VIEW-AS clause.

Several UI fixes/modifications to improve look and feel. Also the tooltip engine has been turned on. Testing.

#23 - 08/31/2015 12:28 PM - Eugenie Lyzenko

Rebased task branch 2606a from P2J trunk revision 10930, new branch revision is 10941.

#24 - 08/31/2015 08:11 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10942.

This update changes the location of the tooltip feature setup. Due to SelectionListGuiIml and SelectionListBodyGuiImpl share the same config(and the same widget ID) there can be confusion if to install tooltip within SelectionListBodyGuiImpl because there is no such object in widget registry, for given ID the registry has SelectionListGuiIml. So the correct way I think is to attach tooltip to the object having correct ID<->implementation mapping in widget registry.

So except the horizontal scrolling this is complete implementation. And I suggest to test and commit the changes if scrolling updates from Hynek will take additional time to be released. What do you think? This way I can shift to the SLIDER widget implementation support. I just worry this widget should be written from scratch and for now I can not even estimate how much time it can take.

#25 - 08/31/2015 08:36 PM - Eugenie Lyzenko

The current testcases for selection-list widget have been uploaded to bzr repo. Version is 1313.

#26 - 09/01/2015 09:08 AM - Greg Shah

Code Review Task Branch 2606a Revision 10942

This is very good! I've made some small cleanups to javadoc and checked that in as revision 10943.

I didn't see anything there for the MAX-CHARS attribute. I think that support still needs to be added.

After Hynek commits 2424c, you can merge to trunk and finish the scrolling work. Ask him questions as needed. Then I'll do a final code review and we will get this into testing.

Good work!

#27 - 09/01/2015 08:49 PM - Eugenie Lyzenko

- File sel_list_p2j_test5_3_0901.jpg added

Task branch 2606a for review updated to revision 10944.

Adding support for painting the disabled state for GUI scrollbar button. The current implementation for button drawing is polygon based. The picture is attached. I'm not sure if it is OK, because it does not 100% like the current 4GL Windows(in bitmap based I guess). What do you think, need to rework or not?

> I didn't see anything there for the MAX-CHARS attribute. I think that support still needs to be added.

Greg, I thought this attribute is for BROWSE, COMBO-BOX and EDITOR widgets according to Progress docs. Do we really need this to be implemented in SELECTION-LIST?

#28 - 09/01/2015 09:13 PM - Eugenie Lyzenko

Testing on 4GL system shows the MAX-CHARS is not queryable/settable attribute for SELECTION-LIST widget. I have used maxWidth variable in SelectionBodyGuiImpl when I started to implement horizontal scrolling before knowing about Hynek's changes. Just left this variable until further implementation.

#29 - 09/02/2015 07:10 AM - Greg Shah

> Do we really need this to be implemented in SELECTION-LIST?

I checked, you're right. We don't need MAX-CHARS.

#30 - 09/02/2015 07:13 AM - Greg Shah

> Adding support for painting the disabled state for GUI scrollbar button. The current implementation for button drawing is polygon based. The picture is attached. I'm not sure if it is OK, because it does not 100% like the current 4GL Windows(in bitmap based I guess). What do you think, need to rework or not?

Yes, it needs to be made 100% correct.

But I wonder if it is bitmap based or if it can be scaled to a larger/smaller size. Please test in the 4GL, if there is a way to make it draw a larger triangle.

Hynek: do you have any ideas on how to make it scale?

#31 - 09/02/2015 07:24 AM - Eugenie Lyzenko

> But I wonder if it is bitmap based or if it can be scaled to a larger/smaller size. Please test in the 4GL, if there is a way to make it draw a larger triangle.

OK. I'll test. As far as I know the scroll buttons is a part of the Windows system controls, not the 4GL elements. So the question will be if the native Windows controls can be scaled or not. In combo-box implementation I used image based approach for drop-down button. The scrollbar buttons/thumb was untouched. Due to polygon based approach the scrollbar buttons can easily be scaled because the triangle coordinates are computing on every painting based on the current button size.

#32 - 09/02/2015 07:34 AM - Greg Shah

> As far as I know the scroll buttons is a part of the Windows system controls, not the 4GL elements. So the question will be if the native Windows controls can be scaled or not.

Yes, but even if they can be scaled in the native WIN32 code, the 4GL may not use that feature.

It is fine to test at the WIN32 level. If it can't be done there, then it is unlikely the 4GL does anything different.

But if it can be scaled at the WIN32 level, we still need to check if the 4GL can cause that change. For example, although the 4GL allows adding/removing scrollbars on certain widgets, I don't know of any way to configure the scrollbar's appearance or size from 4GL code. If it is possible, it would most likely be a side-effect of changing fonts or some other attribute/option that has sizing implications.

If it is not possible, we will gladly move to a bitmapped approach.

#33 - 09/02/2015 07:43 AM - Hynek Cihlar

- File scroll bar icon scaling.png added

Eugenie Lyzenko wrote:

> Task branch 2606a for review updated to revision 10944.
>
> Adding support for painting the disabled state for GUI scrollbar button. The current implementation for button drawing is polygon based. The picture is attached. I'm not sure if it is OK, because it does not 100% like the current 4GL Windows(in bitmap based I guess). What do you think, need to rework or not?

Btw. the scroll bar icons are most likely vector based, they can be fluently scaled in the theme settings. See the attached picture.

#34 - 09/02/2015 08:00 AM - Hynek Cihlar

Greg Shah wrote:

> Adding support for painting the disabled state for GUI scrollbar button. The current implementation for button drawing is polygon based. The picture is attached. I'm not sure if it is OK, because it does not 100% like the current 4GL Windows(in bitmap based I guess). What do you think, need to rework or not?

> Yes, it needs to be made 100% correct.
>
> But I wonder if it is bitmap based or if it can be scaled to a larger/smaller size. Please test in the 4GL, if there is a way to make it draw a larger triangle.
>
> Hynek: do you have any ideas on how to make it scale?

The primary question is if we do need scaling and for what purpose. I think there is no way to set scroll bar size from inside a 4GL program. So the only purpose of scaling would be to provide a usable UI to the user. If we don't implement scaling the scroll bars will be too tiny on high res screens and thus hard to use. I think we do need to scale to provide good user experience.

The second question is how the scaling should behave, i.e. how fluent it needs to be. For example we could create several bitmaps for different resolutions and "jump-scale". I would pick the solution based on the actual requirement here.

#35 - 09/02/2015 08:04 AM - Greg Shah

I can see that some customers may need to customize that value (using themes) for accessibility reasons. We are going to allow control over theme variables as well, so that our customers can duplicate the appearance and accessibility requirements of the original system.

> For example we could create several bitmaps for different resolutions and "jump-scale". I would pick the solution based on the actual requirement here.

The problem we have is that we don't know how customers are going to need to use this. It can't be predicted. If we set arbitrary jump values, then I can guarantee that we will have to revisit this later to add more cases as new customers bring requirements that don't match.

Eugenie: please work to duplicate the exact appearance (including color) using a vector based approach.

#36 - 09/02/2015 08:10 AM - Hynek Cihlar

Greg, there are and likely will be more places where vector based icons/images will be needed. Do you think it would be good idea to use a vector engine and create the graphics in a vector editor?

#37 - 09/02/2015 08:13 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

> Greg, there are and likely will be more places where vector based icons/images will be needed. Do you think it would be good idea to use a vector engine and create the graphics in a vector editor?

Greg, Hynek, I would prefer to avoid this. We can use Java API and knowledge for what we have to do. It is enough. No need to implement another data format import.

#38 - 09/02/2015 08:16 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

> Hynek Cihlar wrote:
>
> > Greg, there are and likely will be more places where vector based icons/images will be needed. Do you think it would be good idea to use a vector engine and create the graphics in a vector editor?
>
> Greg, Hynek, I would prefer to avoid this. We can use Java API and knowledge for what we have to do. It is enough. No need to implement another data format import.

My concern is that hard coding the vector images will lock us down to any future improvements (new themes for example) and the result will vary depending on the quality of each of the hard coded scaling algorithm. In the result we may get a not very appealing UI.

#39 - 09/02/2015 08:19 AM - Greg Shah

Hynek Cihlar wrote:

> Greg, there are and likely will be more places where vector based icons/images will be needed. Do you think it would be good idea to use a vector engine and create the graphics in a vector editor?

Are you suggesting something like this?

https://xmlgraphics.apache.org/batik/tools/rasterizer.html

#40 - 09/02/2015 08:20 AM - Hynek Cihlar

Greg Shah wrote:

> Hynek Cihlar wrote:
>
> > Greg, there are and likely will be more places where vector based icons/images will be needed. Do you think it would be good idea to use a vector engine and create the graphics in a vector editor?
>
> Are you suggesting something like this?
>
> https://xmlgraphics.apache.org/batik/tools/rasterizer.html

Yes. I think Batik would be a good choice.

#41 - 09/02/2015 08:46 AM - Greg Shah

> My concern is that hard coding the vector images will lock us down to any future improvements (new themes for example) and the result will vary depending on the quality of each of the hard coded scaling algorithm. In the result we may get a not very appealing UI.

Either way we still have to exactly duplicate the Windows result. We still have to exactly duplicate result of the scaling at each valid size level.

I don't understand how a 3rd party rasterizer is more likely to duplicate the results pixel-for-pixel than using Java2D.

Java2D has a built-in rasterizer. If we can make it work properly, there are advantages:

1. Fewer project dependencies. Every dependency added is increased security risk and an increased risk of something going functionally wrong at runtime (e.g. if it is missing).
2. The code is easier to understand and maintain.

Even if the SVG rasterizer is really simple to use, there will be parts of the code that would require readers to understand more to deal with it. And in the SVG case, the developers must also learn SVG or at least how to competently use an SVG editor.

We would want to avoid this unless we know that the value of the 3rd party rasterizer is greater than the cost. If we can match the Windows results pixel for pixel using Java2D, then I see that as a better option. More people understand the Java2D code and it is consistent across the project.

Using Batik, we still have to include the SVG files with our project. How is that different than hard coding Java2D calls? If we change them over time, we will have to do a full testing run anyway. They may be data files but our code is so dependent upon them that we will have to maintain them as part of the project.

I guess I'm not fully understanding the value of your proposal.

#42 - 09/02/2015 09:29 AM - Hynek Cihlar

Greg Shah wrote:

> My concern is that hard coding the vector images will lock us down to any future improvements (new themes for example) and the result will vary depending on the quality of each of the hard coded scaling algorithm. In the result we may get a not very appealing UI.

> Either way we still have to exactly duplicate the Windows result. We still have to exactly duplicate result of the scaling at each valid size level.

For more complex shapes, the hard coding approach will diverge from the scale of 1. For example, the window caption buttons look identical at scale one, but the more the scale changes, the more they diverge.

> I don't understand how a 3rd party rasterizer is more likely to duplicate the results pixel-for-pixel than using Java2D.

It is a lot easier to duplicate an existing shape with an SVG editor that hard coding in Java2D. The process is as follows, take a screenshot of the duplicating icon, put it into the SVG editor, draw the vectors on top of the bitmap. You get an exact copy quickly. Now take a relatively simple shape, the X in the window close button. It is not as simple as drawing two lines, it is also the crossing shape and the line endings to get right, and make sure it scales ok.

> Even if the SVG rasterizer is really simple to use, there will be parts of the code that would require readers to understand more to deal with it.

1. The code is easier to understand and maintain.

```
   @Override
   public void draw()
   {
      NativePoint p = physicalLocation();
      final TopLevelWindow<GuiOutputManager> window = topLevelWindow();

      NativeDimension d = physicalDimension();

      updateScale(d);

      final int width = d.width;
      final int height = d.height;

      final NativeRectangle clip = clipRectangle(p, d);

      // draw button
      gd.draw(p, clip, new Runnable()
      {
         @Override
         public void run()
         {
            ColorRgb faceColor = up ? gd.getSysColor(SysColor.COLOR_3DFACE) :
                                      gd.getSysColor(SysColor.COLOR_3DSHADOW);

            gd.setColor(gd.getSysColor(SysColor.COLOR_3DFACE));
            gd.fill3DRect(0, 0, width, height, up);

            gd.setColor(gd.getSysColor(SysColor.COLOR_BTNTEXT));

            // draw icon on button
            switch (type)
            {
               case CLOSE:
                  gd.setLineStroke(LineStroke.SPECIFIED_WIDTH, scaleX(2));

                  // north-west - south-east line
                  gd.drawLine(scaleX(4), scaleY(3), width - scaleX(4), height - scaleY(3));

                  // SW-NE line
                  gd.drawLine(scaleX(4), height - scaleY(3), width - scaleX(4), scaleY(3));

                  // TODO: the SW-NE line is painted with 1/2 thickness of the NW-SE line,
                  //       this unfortunately limits the scaling capabilities
                  gd.drawLine(scaleX(5), height - scaleY(3), width - scaleX(3), scaleY(3));

                  // these rects make the proper line ends
                  gd.setColor(faceColor);
                  gd.fillRect(2, 2, width - 4, scaleY(2));
                  gd.fillRect(2, height - 3, width - 4, scaleY(2));
                  gd.setColor(gd.getSysColor(SysColor.COLOR_BTNTEXT));

                  gd.setLineStroke(LineStroke.DEFAULT);
                  break;

               case ICONIFY:
                  gd.setLineStroke(LineStroke.SPECIFIED_WIDTH, scaleX(2));

                  gd.drawLine(scaleX(5), height - scaleY(4), width - scaleX(7), height - scaleY(4));
                  gd.setLineStroke(LineStroke.DEFAULT);
                  break;

               case MAXIMIZE:
                  if (!window.isMaximized())
                  {
                     drawMaxRect(scaleX(4),
                                 scaleY(3),
                                 width - scaleX(4) - scaleX(5),
                                 height - scaleY(3) - scaleY(4),
                                 false,
                                 faceColor);
```

```
                }
                else
                {
                   // draw the upper window icon

                   drawMaxRect(scaleX(3) + scaleX(3),
                               scaleY(2),
                               width - scaleX(2) - scaleX(3) - scaleX(3) - scaleX(2),
                               height - scaleY(2) - scaleY(2) - scaleY(5),
                               false,
                               faceColor);

                   // draw the lower window icon

                   drawMaxRect(scaleX(3),
                               scaleY(3) + scaleY(3),
                               width - scaleX(2) - scaleX(3) - scaleX(3) - scaleX(2),
                               height - scaleY(2) - scaleY(2) - scaleY(5),
                               true,
                               faceColor);
                }
                break;
             }
          }
       });
   }

   private void drawMaxRect(int      x,
                            int      y,
                            int      width,
                            int      height,
                            boolean  clear,
                            ColorRgb clearColor)
   {
      if (clear)
      {
         gd.setColor(clearColor);
         gd.fillRect(x, y, width, height);
         gd.setColor(gd.getSysColor(SysColor.COLOR_BTNTEXT));
      }

      int strokeSize = scaleX(1);
      gd.setLineStroke(LineStroke.SPECIFIED_WIDTH, strokeSize);

      gd.drawRect(x, y, width, height);

      gd.setLineStroke(LineStroke.SPECIFIED_WIDTH, scaleX(2));
      gd.drawLine(x + 1, y + 1, x + width - strokeSize, y + 1);

      gd.setLineStroke(LineStroke.DEFAULT);
   }

   private void updateScale(NativeDimension size)
   {
      xScale = (double) size.width / BASE_WIDTH;
      yScale = (double) size.height / BASE_HEIGHT;
   }

   private int scaleX(int value)
   {
      return (int) Math.round(xScale * value);
   }

   private int scaleY(int value)
   {
      return (int) Math.round(yScale * value);
   }
```

vs

```
   @Override
   public void draw()
```

```
   {
      NativePoint p = physicalLocation();
      final TopLevelWindow<GuiOutputManager> window = topLevelWindow();

      NativeDimension d = physicalDimension();

      updateScale(d);

      final int width = d.width;
      final int height = d.height;

      final NativeRectangle clip = clipRectangle(p, d);

      // draw button
      gd.draw(p, clip, new Runnable()
      {
         @Override
         public void run()
         {
            ColorRgb faceColor = up ? gd.getSysColor(SysColor.COLOR_3DFACE) :
                                      gd.getSysColor(SysColor.COLOR_3DSHADOW);

            gd.setColor(gd.getSysColor(SysColor.COLOR_3DFACE));
            gd.fill3DRect(0, 0, width, height, up);

            // draw icon on button
            switch (type)
            {
               case CLOSE:
                  gd.drawImage(closeVector.rasterize(x, y, width, height));
                  break;

               case ICONIFY:
                  gd.drawImage(iconifyVector.rasterize(x, y, width, height));
                  break;

               case MAXIMIZE:
                  if (!window.isMaximized())
                  {
                     gd.drawImage(maxVector.rasterize(x, y, width, height));
                  }
                  else
                  {
                     gd.drawImage(unmaxVector.rasterize(x, y, width, height));
                  }
                  break;
            }
         }
      });
   }
```

And in the SVG case, the developers must also learn SVG or at least how to competently use an SVG editor.

From my own experience, this is not any more difficult than using a modern Word processor. It is definitely easier to draw an untrivial shape in an SVG editor than hard code it in Java2D AND make it properly scalable.

#43 - 09/02/2015 09:57 AM - Greg Shah

How confident are you that we can match the Windows vectors pixel for pixel using SVG?

#44 - 09/02/2015 10:03 AM - Greg Shah

Greg Shah wrote:

> How confident are you that we can match the Windows vectors pixel for pixel using SVG?

I'm particularly wondering about whether it will match when scaling is used.

#45 - 09/02/2015 11:54 AM - Hynek Cihlar

Greg Shah wrote:

> How confident are you that we can match the Windows vectors pixel for pixel using SVG?

As much confident as with the hard-coded approach. And I am afraid we won't achieve pixel perfect precision in all cases.

Another problem with the hard coded vector drawing is that our vector primitives in the gui driver interface are limited. These will have to be added and implemented - especially round shapes, curves, gradients, polylines, etc.

#46 - 09/02/2015 11:59 AM - Eugenie Lyzenko

Hynek Cihlar wrote:

> As much confident as with the hard-coded approach. And I am afraid we won't achieve pixel perfect precision in all cases.
>
> Another problem with the hard coded vector drawing is that our vector primitives in the gui driver interface are limited. These will have to be added and implemented - especially round shapes, curves, gradients, polylines, etc.

This can easily be painted by set of horizontal or vertical lines. I guess this is the way the Windows uses. I found the both win32 and 4GL do scaling for scroll button arrows. I'll prepare the samples soon.

#47 - 09/02/2015 12:04 PM - Greg Shah

> As much confident as with the hard-coded approach.

Eugenie: please give the SVG approach a try. Don't worry about the scaling right now, just see if you can get the min/max/close buttons and the up/down/left/right arrows replaced with an SVG rasterized bitmap.

Another advantage of this approach is that the result is a bitmap that can get cached on the web client side and thus the result may be very slightly faster.

> And I am afraid we won't achieve pixel perfect precision in all cases.

This is something we do need to achieve. If existing vector tools don't match, we may have to augment that with more extreme measures.

Automated testing alone is going to make this very important.

> our vector primitives in the gui driver interface are limited. These will have to be added and implemented - especially round shapes, curves, gradients, polylines, etc.

I had expected the need for circles (radio-set). But otherwise, I wasn't aware of other specific needs.

We can and will add primitives as needed. Adding a primitive is the matter of some hours or maybe a day or two at maximum.

#48 - 09/02/2015 12:49 PM - Hynek Cihlar

Greg Shah wrote:

> > As much confident as with the hard-coded approach.
>
> Eugenie: please give the SVG approach a try. Don't worry about the scaling right now, just see if you can get the min/max/close buttons and the up/down/left/right arrows replaced with an SVG rasterized bitmap.
>
> Another advantage of this approach is that the result is a bitmap that can get cached on the web client side and thus the result may be very

> slightly faster.
>
> > And I am afraid we won't achieve pixel perfect precision in all cases.
>
> This is something we do need to achieve. If existing vector tools don't match, we may have to augment that with more extreme measures.
>
> Automated testing alone is going to make this very important.
>
> > our vector primitives in the gui driver interface are limited. These will have to be added and implemented - especially round shapes, curves, gradients, polylines, etc.
>
> I had expected the need for circles (radio-set). But otherwise, I wasn't aware of other specific needs.

The set of required primitives will depend on the supported Windows themes. In general the older ones are more simple than the new ones. 4GL reference docs show screenshots from different versions of Windows and their themes, including a theme which looks like the default theme of Windows 2000. This one will already require more complex drawings - it contains gradients, lines with different tips, differently rounded corners, '?' in one of the window caption buttons, etc.

Btw. radio button circle is drawn with two arcs to achieve the shadow effect.

#49 - 09/02/2015 01:01 PM - Greg Shah

> 4GL reference docs show screenshots from different versions of Windows and their themes, including a theme which looks like the default theme of Windows 2000. This one will already require more complex drawings - it contains gradients, lines with different tips, differently rounded corners, '?' in one of the window caption buttons, etc.

I have no plans to implement a Windows 2000 theme.

I expect the following would be most useful:

1. Windows 7
2. Windows "Classic" (not really a theme but corresponds to the desktop you get on Windows Server 2008 in Terminal Services or Citrix, thus it may be a common thing for "thin client" customers to see)
3. Windows XP (not a theme, but would require duplication of the widget-level Aero appearance)

I only plan for 1 and 2 as part of our current work. If any other customers want something else, they can pay us to add it.

> Btw. radio button circle is drawn with two arcs to achieve the shadow effect.

Good information, thanks.

#50 - 09/02/2015 01:43 PM - Hynek Cihlar

- File windows 7 basic theme.png added

Greg Shah wrote:

> > 4GL reference docs show screenshots from different versions of Windows and their themes, including a theme which looks like the default theme of Windows 2000. This one will already require more complex drawings - it contains gradients, lines with different tips, differently rounded corners, '?' in one of the window caption buttons, etc.
>
> I have no plans to implement a Windows 2000 theme.
>
> I expect the following would be most useful:
>
> 1. Windows 7
> 2. Windows "Classic" (not really a theme but corresponds to the desktop you get on Windows Server 2008 in Terminal Services or Citrix, thus it may be a common thing for "thin client" customers to see)
> 3. Windows XP (not a theme, but would require duplication of the widget-level Aero appearance)
>
> I only plan for 1 and 2 as part of our current work. If any other customers want something else, they can pay us to add it.

For Windows 7 I assume you mean the default non-aero Windows 7 Basic theme? See the screenshot.

#51 - 09/02/2015 01:52 PM - Eugenie Lyzenko

Hynek, the 1.7 batik binary is enough to use this rasterizer in P2J? Do you have any sample Java code that demonstrates how it works with SVG image?

#52 - 09/02/2015 02:08 PM - Hynek Cihlar

Eugenie Lyzenko wrote:

> Hynek, the 1.7 batik binary is enough to use this rasterizer in P2J? Do you have any sample Java code that demonstrates how it works with SVG image?

Eugenie, 1.8 seems to be the latest stable. See the samples here for example, http://www.programcreek.com/java-api-examples/index.php?api=org.apache.batik.apps.rasterizer.SVGConverter.

#53 - 09/02/2015 02:56 PM - Greg Shah

> For Windows 7 I assume you mean the default non-aero Windows 7 Basic theme?

Yes.

For now, we should focus on the windows classic that is visible in our 4GL test env.

We will schedule development of more complete theming support (including the Windows 7 Basic theme) for later.

#54 - 09/02/2015 03:20 PM - Eugenie Lyzenko

What I can tell from samples is the basic approach we can use:

1. Having SVG files for every image we want to be scaled(one set per one theme).
2. When it is time to draw it we can identify the image with name and required size.
3. If the required raster image not in cache, use SVGConverter class to open SVG source and prepare say PNG image to draw. Batik tool uses file system to put the result, I think we can use in memory file to store the image and put into cache.
4. Then we can take image from cache and draw it.

Notes:

1. Everything will be happen on the client side.
2. When we need the same image but size is different - we have to create new raster data from initial SVG - this will another file in memory. Or we have to store in memory all SVGConverter classes for every images we want to deal with. Otherwise the usage of this tool become under the doubt.

Let me know if I've missed for something.

#55 - 09/02/2015 03:35 PM - Greg Shah

> Let me know if I've missed for something.

Your assumptions seem reasonable. Please make sure the SVG files are read from the p2j.jar like other essential resources.

> When we need the same image but size is different - we have to create new raster data from initial SVG - this will another file in memory.

Yes, this is fine.

#56 - 09/02/2015 04:02 PM - Hynek Cihlar

Eugenie, also try to keep only unique set of converted images. For example when multiple scroll bars are painted we only need to reference two pieces of data (source images) representing the up and down buttons.

Also I wouldn't use png but a raw bitmap format to avoid additional processing when widgets are redrawn. Note that this may happen frequently especially during operations like resizing and scrolling.

#57 - 09/02/2015 04:07 PM - Greg Shah

Sergey: please review this task history from note 30 to the end.

Eugenie/Hynek: Sergey has implemented an image caching implementation in 1811q, but it is implemented in the GUI web driver.

I am wondering if there is some value in moving any of the image caching implementation into common code.

#58 - 09/02/2015 10:35 PM - Eugenie Lyzenko

Another issue to solve in the SVG converter implementation. The converter takes the source file as name or URL while the data given from p2j.jar file is InputStream type. So we need one more step to create another temporary file from InputStream to pass as option to SVG converter.

#59 - 09/03/2015 02:45 AM - Sergey Ivanovskiy

Batik uses asynchronous renderers and alarms the result is ready or not. It has very complicated implementation of dynamic svg. If we need only static svg, I advise to use another library, for example https://svgsalamander.java.net/ or http://www.jfree.org/jfreesvg/ if you would not like to sink in the Batik code.

#60 - 09/03/2015 02:51 AM - Hynek Cihlar

Eugenie Lyzenko wrote:

> Another issue to solve in the SVG converter implementation. The converter takes the source file as name or URL while the data given from p2j.jar file is InputStream type. So we need one more step to create another temporary file from InputStream to pass as option to SVG converter.

Eugenie, you will probably have to use the transocding API directly. The ImageTranscoder seems to be the transcoder to use. See this page, http://people.apache.org/~clay/batik/rasterizerTutorial.html.

#61 - 09/03/2015 02:55 AM - Hynek Cihlar

Sergey Ivanovskiy wrote:

> Batik uses asynchronous renderers and alarms the result is ready or not.

This seems to be true for the rendering part, JSVGCanvas. For this task we will use only its transcoding capability.

#62 - 09/03/2015 06:33 PM - Eugenie Lyzenko

I'm going to put the SVG images into

```
../ui/client/gui/theme
```

directory inside the sources/p2j.jar. In this directory for each theme will be created single subdir, currently only classic will exists.

Is this OK location?

#63 - 09/03/2015 11:49 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10945.

Adding basic steps to implement SVG handler. Still does not work causing transcoder exception. Continue working.

#64 - 09/04/2015 09:34 AM - Greg Shah

Eugenie Lyzenko wrote:

> I'm going to put the SVG images into
> [...]
> directory inside the sources/p2j.jar. In this directory for each theme will be created single subdir, currently only classic will exists.
>
> Is this OK location?

Yes, this is a good plan.

#65 - 09/04/2015 04:57 PM - Eugenie Lyzenko

Rebased task branch 2606a from P2J trunk revision 10931, new branch revision is 10945.

The SVG code/libs has been moved to another branch - 2678a.

#66 - 09/09/2015 11:19 AM - Eugenie Lyzenko

Rebased task branch 2606a from P2J trunk revision 10933, new branch revision is 10948.

#67 - 09/09/2015 04:38 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10949.

Please review the horizontal scrolling implementation for mouse and keyboard. Some key processing issue has been also fixed. Cleanup and TODO commented.

#68 - 09/09/2015 05:16 PM - Greg Shah

Code Review Task Branch 2606a Revision 10949

I'm good with the changes.

Please get this regression tested (conversion and runtime). If it passes, please merge to trunk. If there are problems, fix them and I will review.

#69 - 09/09/2015 05:25 PM - Eugenie Lyzenko

Greg Shah wrote:

> Code Review Task Branch 2606a Revision 10949
>
> I'm good with the changes.
>
> Please get this regression tested (conversion and runtime). If it passes, please merge to trunk. If there are problems, fix them and I will review.

OK. The regression testing has been started.

#70 - 09/09/2015 07:46 PM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10950.

Please review this small GUI fix for dotted focus marker rectangle. This is pure GUI change, no additional conversion test required. I would like to include this change into current testing/merge process if there are no objections.

#71 - 09/09/2015 08:48 PM - Eugenie Lyzenko

Conversion test completed without regressions. The generated codes are the same. Starting runtime testing for 10950 revision.

#72 - 09/10/2015 07:53 AM - Greg Shah

Code Review Task Branch 2606a Revision 10950

My only question is regarding this:

```
            // inside this call widget origin is always (0,0)
            NativePoint po = new NativePoint(0, 1);
```

The comment is confusing. Please clarify it.

> This is pure GUI change, no additional conversion test required. I would like to include this change into current testing/merge process if there are no objections.

No objections.

#73 - 09/10/2015 08:06 AM - Eugenie Lyzenko

Greg Shah wrote:

> Code Review Task Branch 2606a Revision 10950
>
> My only question is regarding this:
>
> [...]
>
> The comment is confusing. Please clarify it.

Yes, I should change the comment. One pixel Y shit is to consider internal black frame rectangle. If we use (0,0) the dotted frame for selected item is corrupted(overridden by body frame).

The correct version is:

```
...
            // inside this call widget origin is (0,1) to consider internal border
            NativePoint po = new NativePoint(0, 1);
...
```

#74 - 09/10/2015 08:22 AM - Greg Shah

Please commit that comment change.

#75 - 09/10/2015 09:52 AM - Eugenie Lyzenko

Task branch 2606a for review updated to revision 10951.

> Please commit that comment change.

Done.

#76 - 09/10/2015 10:13 AM - Greg Shah

Good, this can be merged to trunk if it passes testing. When do you expect runtime testing results?

#77 - 09/10/2015 11:29 AM - Eugenie Lyzenko

> Good, this can be merged to trunk if it passes testing. When do you expect runtime testing results?

The CTRL-C testing completed without regressions. The main part was 2 cycles with remaining tests: TC-JOB-CLOCK-004 and TC-JOB-CLOCK-005. So I need at least one more round to be sure(I think they are both OK).

I guess today in a 4-5 hours the update will be ready to merge.

#78 - 09/10/2015 04:43 PM - Eugenie Lyzenko

Regression testing completed. Results: 2606a_10950_16cd2a2_20150910_evl.zip. No problems found.

My plan is:
Rebase 2606a with 2678a(10934) and because only GUI code will be touched(only ui/client/gui/ScrollBarGuiButton) - merge final 2606a into trunk.

Is it OK?

#79 - 09/10/2015 04:48 PM - Greg Shah

> because only GUI code will be touched(only ui/client/gui/ScrollBarGuiButton) - merge final 2606a into trunk.
>
> Is it OK?

Yes, please do so.

#80 - 09/10/2015 05:15 PM - Eugenie Lyzenko

Task branch 2606a has been updated and rebased with 10934 up to revision 10953. Starting merge into trunk.

#81 - 09/10/2015 05:40 PM - Eugenie Lyzenko

Task branch 2606a has been merged/committed into trunk as bzr revision 10935.

#82 - 09/10/2015 05:51 PM - Eugenie Lyzenko

Task branch 2606a has been archived.

#83 - 09/10/2015 06:12 PM - Greg Shah

- Status changed from New to Closed

#84 - 11/16/2016 12:13 PM - Greg Shah

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

Files

sel_list_p2j_test0_0820.jpg	45.4 KB	08/21/2015	Eugenie Lyzenko
sel_list_p2j_test1_0821.jpg	50.5 KB	08/21/2015	Eugenie Lyzenko
sel_list_test5_4gl_1.jpg	58.8 KB	08/29/2015	Eugenie Lyzenko
sel_list_p2j_test5_0828.jpg	44.1 KB	08/29/2015	Eugenie Lyzenko
sel_list_test5_3_4gl_0.jpg	70.1 KB	08/29/2015	Eugenie Lyzenko
sel_list_p2j_test5_3_0828.jpg	46.8 KB	08/29/2015	Eugenie Lyzenko
sel_list_p2j_test5_3_0901.jpg	46.8 KB	09/02/2015	Eugenie Lyzenko
scroll bar icon scaling.png	25.6 KB	09/02/2015	Hynek Cihlar
windows 7 basic theme.png	86.9 KB	09/02/2015	Hynek Cihlar