User Interface - Feature #2334

Feature # 2252 (Closed): implement GUI client support

implement the GUI message line widget

07/09/2014 11:54 AM - Greg Shah

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Marius Gligor	**% Done:**	100%
Category:		**Estimated time:**	40.00 hours
Target version:	GUI Support for a Complex ADM2 App		
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 09/17/2014 01:42 AM - Marius Gligor

- Status changed from New to WIP

- Assignee set to Marius Gligor

#2 - 09/17/2014 07:29 AM - Marius Gligor

- File message-area3.png added

- File message-area1.png added

- File message-area2.png added

In OpenEdge GUI the message area is a reserved region at the bottom of window above status bar implemented as a scrollable list of strings widget.
The widget has 2 lines always visible. The function MESSAGE-LINES return the number of visible lines.
According to my tests it returns always 2 regarding the number of lines available in the list at one moment.
I didn't found any way to change the number of visible lines of message area.
A vertical scroll bar is displayed only when the number of lines is greater than the number of visible lines (2)
No horizontal scroll bar is used. If the text exceed the length the could be displayed in one line the text is wrapped to the next line.
According to my tests the maximum text length in one line is not fixed because the font used by message area is proportional.
When the main window is resized message area size is resized too but nothing happen with text in lines. (see attached pictures)

Before to implement a scrollable container I started to design and implement a GUI scroll bar widget. In ChUI no such widget is implemented.

#3 - 09/23/2014 01:33 PM - Marius Gligor

- File mag_upd20140923a.zip added

- File gui_message_area.png added

A first implementation of GUI message area. I implemented a scroll bar widget a scroll pane widget and a scrollable list widget.
Then I implemented the GUI message area as a scroll pane having a scrollable list to hold messages (see attached picture).
Some problems that I found so far:

1. The implementation of GUI widgets still depends on coordinate conversion between pixels units and character units.

2. Mouse and keyboard support should be added to widgets.

3. Unfortunately at this point we cannot test the GUI client because is not yet properly implemented.

In conclusion they are still lot of works to do after solving the problems from 1,2 and 3.
Nevertheless the basic implementation is here.

#4 - 09/23/2014 06:07 PM - Greg Shah

Code Review 0923a

1. In WindowGuiImpl do we need to keep the hard coded message output (e.g. line 1...)? Even with the limited GUI client support it seems that a message statement on the default window could be made to work. If not, what are the specific inhibitors? The obvious thing missing is that MessageAreaImpl is not connected.

2. If I understand correctly, the full list processing of the message area is not there yet. Have you answered the list processing questions noted in #2252 notes 79 and 212? You can go further with the list processing in this first pass without other GUI support, right?

3. What is the idea behind ScrollBar.Bars.BOTH? It seems like it would be easier to implement separate instances (one HORIZONTAL and one VERTICAL) than to implement a single instance that draws in two places.

4. I think it makes sense to explain in javadoc that for VERTICAL scrollbars, ScrollBar.Position.LEFT is the "up" and ScrollBar.Position.RIGHT is "down".

5. Why not put the scroll listener notification code into ScrollBar instead of ScrollBarGuiImpl?

6. MessageLineGui needs a history entry.

7. ScrollableListGuiImpl and MessageAreaGuiImpl need standard headers and class javadoc.

#5 - 09/23/2014 06:13 PM - Greg Shah

1. The implementation of GUI widgets still depends on coordinate conversion between pixels units and character units.

Yes, understood.

2. Mouse and keyboard support should be added to widgets.

Yes.

3. Unfortunately at this point we cannot test the GUI client because is not yet properly implemented.

But at this point we can at least get the STATUS and MESSAGE statements working with the DEFAULT-WINDOW. Or is there something broken that doesn't allow that?

I am suggesting that you would not use ask.p since that requires frame support. But you can run some very simple code like:

```
MESSAGE "Hello World!".
```

Message and status statements don't rely upon frames or other widgets. So long as the ThinClient methods are connected to the backing GUI implementation it should work.

#6 - 09/25/2014 11:29 AM - Marius Gligor

- File gui-client.png added

- File mag_upd20140925a.zip added

Here are my latest changes. I fixed many problems related to status bar and message area widgets but is still an intermediate work.

1. First and most important I fixed a blocked bug in GuiOutputManager#processDefaultWindow which was the cause of creating multiple instances of the same window.
Now the client work properly please observe the status bar message in the attached picture. :-)

2. I fixed also many other drawing issues. However they are still works to do especially on message area widget which for GUI is a complex widget.
In fact almost all GUI widgets are more complex than related ChUI widgets.

3. I tried to inherit ASAP from ChUI already existing code. Nevertheless some changes might be necessary.

4. The attached changes are still using the old style of coordinates.
On the next step I have to update and merge my changes with the new Hynek updates related to decimal to integer coordinates conversion.
Then I have to fix all the aspects of status line and message area related to coordinates.
Seems to be a difficult task because I expect many errors and conflicts after merge but finally should works.

#7 - 09/25/2014 12:30 PM - Greg Shah

Code Review 0925a

This is really good! I'm excited that you have achieved the "Hello World" for GUI.

My only comment is that I see at least 2 pieces of code that will duplicated in every widget, but which probably could be common code: repaint() and the draw() code that detects if the font has changed and forces a relayout of the containing window. There may be more code than this, but it seems like all widgets will have this in common. How do we most effectively move such code to a common location when all GUI widgets don't have a single common GUI-specific parent (their common parents are not specific to ChUI or GUI)? GuiPrimitives is too low-level for that code and GuiScreenDriver is too far away from widget-level concerns. Perhaps GuiOutputManager is a good location?

Or can the common features be refactored to be truly common to both ChUI and GUI (although the font changing logic definitely only affects GUI)?

#8 - 09/25/2014 12:40 PM - Marius Gligor

Yes we could do the code more abstract. But first of all I want to have the coordinates fixed.
Fonts are used only on GUI clients because FONT-TABLE and COLOR-TABLE are available only for GUI interfaces.
Also both fonts and colors could be changed dynamically and should be reevaluate whenever the widgets are draw.
For efficiency here we have first to check if they are really changed and reevaluate if and only if has been changed.

#9 - 09/25/2014 01:10 PM - Greg Shah

Agreed and understood.

#10 - 09/30/2014 10:56 AM - Marius Gligor

- File message-area5.png added

- File message-area4.png added

- File message-area6.png added

- File message-area7.png added

- File mag_upd20140930a.zip added

1. For ChUI the number of message lines is hard coded and the ThinClient# getMessageLines return always Window.RESERVED_LINES - 1 which currently is 2.
For GUI I didn't found any way to change the number of message area lines. Should be the same value as for ChUI or should be a parameter stored in server directory?

2. In order to receive mouse events we have to set the state value to true in MouseHandler class. I did that in GuiOutputManager. processDefaultWindow by adding the following statement: sim.captureMouseEvents(true) The question here is:
The mouse enabled should be a parameter in WindowConfig? Who decides if a window has mouse or not?

3. Regarding selection of mouse source which is the widget that should handle the mouse events. Basically we have widgets and containers. Containers should holds other widgets or containers, widgets not. Container widgets coordinates are relative to container while mouse position is relative to window which is the ancestor container of window widgets. I implemented a recursive algorithm to find mouse source (see AbstractContainer#findMouseSource). The algorithm works fine but not for WindowTitleBar which is not yet properly implemented is just a dummy implementation. I think that on the next task we have to provide an appropriate implementation for WindowTitleBar which shall use the same algorithm to find mouse sources. Another issue that should be fixed here is to implement an algorithm to select the right widget when widgets are overlapped perhaps using the Z order.

4. Some times we need to know on which area of the mouse source widget surface is the mouse cursor position. I implemented an algorithm to translate the mouse pointer coordinates to widgets relative (see AbstractWidget.translate) coordinates. For example I used this algorithm in ScrollableListGuiImpl to calculate which item in the list was clicked by mouse.

5. Regarding mouse click I think that we have to add more code on handlers like which button was clicked, if double click etc.

6. Regarding drawing the scroll bar widgets is a complex widgets from the drawing point of view. So far I implement almost all drawing aspects. What should be still fixed is to set a minimum size for thumb button. The size of thumb button is calculated and this size could be result small even zero so we have define a minimum size.

7. I inherited as much as possible from the current ChUI widgets implementation. I think we need a mechanism to communicate between widgets. I know that we already have an event based system which allow to send notifications to registered listeners but my idea is to design a direct communication mechanism like a direct call for example: widget.sendMessage(Message message)

8. Even thought the widgets will suffer more changes in the future the message area could be considered actually as implemented. I think that we have to keep the same paradigm when we have to create new widgets. A complex widgets should be build from basic widgets. For example the WindowTitleBar should be designed as a container having an icon, a system menu and 3 buttons. Also system menu should be designed as a pop-up menu which in turn is a specialized menu widget. Menu widgets are containers containing menu items.

#11 - 09/30/2014 01:00 PM - Greg Shah

Should be the same value as for ChUI or should be a parameter stored in server directory?

As I noted in note 79 of #2252, I previously found no way to change the number of message lines, whether GUI or ChUI. There is no reason to have this be parameterized, when Progress itself hard codes it. Just fix this as the same value for both GUI and ChUI.

The mouse enabled should be a parameter in WindowConfig? Who decides if a window has mouse or not?

Have you found a configurable way for the 4GL developer to specify this under program control? If not, then this is OK to do as part of the GUI-specific code. Although we do know that there is some limited possible ChUI mouse support, we have not yet seen a customer that actually uses it. So for now, having the GUI code enable mouse is fine.

The algorithm works fine but not for WindowTitleBar which is not yet properly implemented is just a dummy implementation. I think that on the next task we have to provide an appropriate implementation for WindowTitleBar which shall use the same algorithm to find mouse sources.

Understood. This will have to be part of #2340. Please add an entry to that task to explain this requirement.

Another issue that should be fixed here is to implement an algorithm to select the right widget when widgets are overlapped perhaps using the Z order.

Yes. This probably needs to be done as part of #2370. Please add an entry to that task to explain this requirement.

Regarding mouse click I think that we have to add more code on handlers like which button was clicked, if double click etc.

I'm not sure about this. When it gets to the 4GL widget mouse-specific behavior, we will start to need the 4GL event system to be mouse-aware. In other words, double clicking, choose and other 4GL events can be generated by mouse processing and this common support is probably not going to be implemented in widget-specific handlers.

What should be still fixed is to set a minimum size for thumb button. The size of thumb button is calculated and this size could be result small even zero so we have define a minimum size.

OK. Please do fix this.

I inherited as much as possible from the current ChUI widgets implementation. I think we need a mechanism to communicate between widgets. I know that we already have an event based system which allow to send notifications to registered listeners but my idea is to design a direct communication mechanism like a direct call for example: widget.sendMessage(Message message)

I don't understand the requirement here. I don't know of any case where one widget needs to communicate to another widget. For the MESSAGE statement, the input will be coming from the server, through ThinClient and then passed to the containing Window. That window instance would be responsible for routing it to the contained message area. Please see Vadim's work in #2229 where he is very far along in implementing this for the MESSAGE statement.

Do you have other examples besides MESSAGE or STATUS? Both of these would be handled the same way. Is there some general case of widget to widget communication?

Even thought the widgets will suffer more changes in the future the message area could be considered actually as implemented. I think that we have to keep the same paradigm when we have to create new widgets. A complex widgets should be build from basic widgets. For example the WindowTitleBar should be designed as a container having an icon, a system menu and 3 buttons. Also system menu should be designed as a pop-up menu which in turn is a specialized menu widget. Menu widgets are containers containing menu items.

I generally agree with this. However, we need to be very clear in the boundary between a 4GL widget and these sub-widget components.

#12 - 09/30/2014 01:05 PM - Greg Shah

Why do the message-area5.png, message-area6.png and message-area7.png show a message area that is 8 lines in size? The 4GL GUI doesn't change size like that as far as I know. The list is always only 2 lines tall, the scrollbar never has a thumb and the list is scrolled line by line using the up/down arrow buttons. Were your examples just for testing?

#13 - 09/30/2014 01:37 PM - Marius Gligor

The picture are uploaded just to show how scroll bar looks. I forced 8 lines instead 2 for message area.
When message area has only 2 visible lines the scroll bar is displayed only when the number of lines stored on message area is greater than 2 and only buttons are displayed like in message-area4.png because the height of the message area does not allow drawing other parts of scroll bar.

#14 - 09/30/2014 01:43 PM - Greg Shah

Code Review 0930a

This looks really good.

1. There are still some hard coded message lines left behind.

2. If I understand correctly, the list management features are not yet compatible with the 4GL. In other words, there seems to be no max list size and the dropping of old messages is not implemented.

3. Please post answers to each of the open questions listed in note 79 of #2252. I would like to make sure that we fully understand how this should work.

4. Does the MessageAreaGuiImpl.split() method exactly match the splitting behavior of the 4GL?

#15 - 09/30/2014 01:46 PM - Constantin Asofiei

Greg/Marius: in case you are not aware, see note 3 from #2252 - the message area in GUI holds a maximum of 50 messages. At the time I was investigating this, I couldn't find a way to configure this value.

#16 - 09/30/2014 03:28 PM - Marius Gligor

Thanks Constantin I will check on customer's server.
I have another question related to fonts please: The call to FontManager.getTextWidth require font/text metrics.
I saw 2 files in my p2j project font-metrics.dtd and text-metrics.dtd but no xml files.
Where I could find examples on how to use font and text metrics on my p2j project?

#17 - 10/01/2014 06:03 AM - Marius Gligor

I'm trying to test the following p4 statement current-window:message-area-font=5. which in turn is converted in Java as currentWindow().unwrapWindow().setMessageAreaFont(new integer(5)); but it does not work for GUI clients.
Doing debugs I found why. When the window configuration, is pushed down to client WindowConfig.applyConfig is called but here is the problem:

I'm struggle to understand this snippet of code from WindowConfig class

```
public void applyConfig(ComponentConfig config)
{
   // a hack to not override client-side settings, for GUI mode
   if (OutputManager.getDriver().isChui())
   {
      super.applyConfig(config);

      if (config instanceof WindowConfig)
      {
         WindowConfig other = (WindowConfig) config;

         setStatusAreaFont(other.statusAreaFont);
         setMessageAreaFont(other.messageAreaFont);
         setHelp(other.help);
         setTitleFont(other.getTitleFont());

         // TODO: which other data needs to be re-applied?
      }
   }
   else
   {
      setTitle(((WindowConfig) config).getTitle());
   }
}
```

In class history I found: Added GUI-only hack to allow the window title to be pushed from the server-side to client-side, without overriding other config.
But: setMessageAreaFont(other.messageAreaFont) for example should works on GUI only and doesn't work because the condition to be a ChUI client.
What is in fact the logic that should be implemented here?

#18 - 10/01/2014 08:29 AM - Greg Shah

I'm sure that any such code is just temporary and has no deep meaning. You can remove the conditional "protection" logic and see what happens.

#19 - 10/01/2014 11:58 AM - Marius Gligor

- File mag_upd20141001a.zip added

1. The number of visible lines is Window.RESERVED_LINES - 1 (2) like on ChUI .

2. Message area could hold maximum 50 message lines. I tested on customer's server and is true.
I implemented and tested the dropping of old messages.

3. I did some changes to scroll bar widget to assign a default size to thumb button when computed size is to small.

4. Related to long messages that should be split the alghoritm is different for ChUI and for GUI (see my note #2)
ChUI messages are split in ThinClient so I did a small change to allow the message split only for ChUI in ThinClinet.
GUI messages are split in MessageAreaGuiImpl when messages are added.
I redesigned the algorithm to split long messages in MessageAreaGuiImpl. Unfortunately I tested the algorithm using character units only.
The implementation require font metrics. I asked how to add font metrics to project but I receive no response so far.

5. Also I need to configure a font table having 2 or 3 entries in server directory in order to test a message-area-font change and window layout.

6. I did also some improvements on message area implementation.

#20 - 10/01/2014 06:26 PM - Greg Shah

Code Review 1001a

The code changes look good.

I think the ThinClient.splitMessage() should probably be moved into the ChUI message impl. Let the TC code be common and delegate the different splitting algorithms to the ChUI/GUI specific widget classes.

Constantin will have to answer the font questions, but generally the font configuration must be captured from the original Windows system. Then we load it at runtime and use those metrics to duplicate the 4GL rendering.

#21 - 10/06/2014 11:30 AM - Marius Gligor

- File mag_upd20141006a.zip added

I updated and configured my project to use AOP builder. Here are my latest changes and fixes that I did after doing the merge with Constantin updates.

1. I tested the message split algorithm for GUI implementation. For Swing GUI the AWT FontMetrics is used to measure the length of the text. (see SwingGuiDriver,getTextWidth and SwingGuiPrimitives.getTextWidth) No other configurations seems to be mandatory in order to measure a text width in native units (pixels) for a given font.

2. Currently I'm thinking for a solution to move ThinClient.splitMessage() and make TC code common.
Unfortunately much of the code inside ThinClient is ChUI specific which make the separation a little bit more difficult.

#22 - 10/06/2014 11:57 AM - Greg Shah

Code Review 1006a

Everything looks good except there is some hard coded message output in MessageAreaImpl.draw().

> Currently I'm thinking for a solution to move ThinClient.splitMessage() and make TC code common. Unfortunately much of the code inside ThinClient is ChUI specific which make the separation a little bit more difficult.

Did you look at Vadim's work in #2229? As part of that work, he has moved the splitMessage() implementation into Window. Make sure you don't duplicate his work. The vig_upd20141006a.zip in #2252 has his latest work, but it is not merged with Constantin's changes yet.

Besides this splitMessage() improvement, what else is left to do?

#23 - 10/06/2014 12:09 PM - Marius Gligor

MessageAreaImpl in package com.goldencode.p2j.ui.client.gui is no longer used on GUI. The MessageAreaGuiImpl is used for GUI.
Here is another problem because the MessageAreaGuiImpl cannot extends from MessageArea like MessageAreaImpl and I have to check for a solution to eliminate the MessageAreaImpl for GUI.

Apart of this problem I think no other works left to do on this task.

#24 - 10/07/2014 06:17 AM - Marius Gligor

- File mag_upd20141007a.zip added

Here are my latest changes which contains:

1. I fixed MessageAreaImpl issue. I renamed MessageArea class as AbstractMessageArea. Form this class I extracted an interface MessageArea which is implemented also by MessageAreaGuiImpl. The class com.goldencode.p2j.ui.client.gui.MessageAreaImpl is no longer needed and was deleted from project.

2. The MessageLineGui widget is used to display INSERT indicator. In GUI seems to have no such indicator but nevertheless I will find if is true or not working on the next task #2415
If I shall find such a feature for GUI I have to provide properly implementation.

3. I did a small change inside ThinClient which allow messages to work properly on GUI. Finally this part should be fixed after task #2229 will be finished. I think that we have to override some Window methods in WindowGuiImpl and provide GUI specific implementation.

If you agree my changes we could go forward to put the changes into regression tests and distribute.
When task #2229 will be finished I have to do the appropriate changes on GUI message area implementation.

#25 - 10/07/2014 09:50 AM - Greg Shah

Code Review 1007a

client/chui/MessageAreaImpl.java is missing a history entry. Otherwise I am fine with the update. Please start runtime regression testing.

Even if #2415 finds that there is a insert indicator, I don't want to keep MessageLineGui as the widget name that provides such support. It is too confusing since it really seems like this should be a part of the message support. So, any subsequent insert indicator support should probably be named InsertIndicator.java.

> When task #2229 will be finished I have to do the appropriate changes on GUI message area implementation.

Understood.

#26 - 10/07/2014 10:11 AM - Marius Gligor

- File mag_upd20141007b.zip added

Fixed Code Review 1007a. I renamed MessageLineGui to InsertIndicator.java
I'm going to start regression tests.

#27 - 10/07/2014 10:34 AM - Greg Shah

Code Review 1007b

I'm good with the update.

#28 - 10/08/2014 02:47 AM - Marius Gligor

- Status changed from WIP to Review

- % Done changed from 0 to 90

1007b passed regression tests. committed revision 10621.
This task require more changes after task #2229 will be finished.

#29 - 03/02/2015 02:24 PM - Greg Shah

- % Done changed from 90 to 100

- Status changed from Review to Closed

#30 - 11/16/2016 12:12 PM - Greg Shah

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

Files

message-area1.png	15.7 KB	09/17/2014	Marius Gligor
message-area2.png	22.5 KB	09/17/2014	Marius Gligor

message-area3.png	21.7 KB	09/17/2014	Marius Gligor
mag_upd20140923a.zip	47.8 KB	09/23/2014	Marius Gligor
gui_message_area.png	6.97 KB	09/23/2014	Marius Gligor
gui-client.png	7.04 KB	09/25/2014	Marius Gligor
mag_upd20140925a.zip	69.4 KB	09/25/2014	Marius Gligor
message-area4.png	3.24 KB	09/30/2014	Marius Gligor
message-area5.png	4.9 KB	09/30/2014	Marius Gligor
message-area6.png	4.85 KB	09/30/2014	Marius Gligor
message-area7.png	4.89 KB	09/30/2014	Marius Gligor
mag_upd20140930a.zip	83.6 KB	09/30/2014	Marius Gligor
mag_upd20141001a.zip	208 KB	10/01/2014	Marius Gligor
mag_upd20141006a.zip	220 KB	10/06/2014	Marius Gligor
mag_upd20141007a.zip	223 KB	10/07/2014	Marius Gligor
mag_upd20141007b.zip	223 KB	10/07/2014	Marius Gligor