User Interface - Feature #2478

Feature # 2252 (Closed): implement GUI client support

Feature # 2446 (Closed): implement BUTTON and IMAGE GUI widgets (runtime and conversion support)

button image support

01/06/2015 05:49 PM - Greg Shah

Status:	Closed	**Start date:**	
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	**version:**	
vendor_id:	GCD		

Description

History

#1 - 01/06/2015 05:50 PM - Greg Shah

Image handling methods for buttons:

LOAD-MAGE-UP
LOAD-IMAGE-DOWN
LOAD-IMAGE-INSENSITIVE

Reproduce native 4GL behavior for loading/using images in buttons. Including dynamical loading.

#2 - 01/07/2015 07:03 PM - Eugenie Lyzenko

The update evl_upd20150107a.zip has passed the main runtime regression cycle. Waiting for CTRL-C part to be completed.

#3 - 01/07/2015 08:37 PM - Eugenie Lyzenko

Runtime testing completed. The results - 10696_65fcb0a_20150107_evl.zip. The 3-way tests was started in separate session.

The update is ready to be committed and distributed.

#4 - 01/07/2015 10:42 PM - Greg Shah

Good, go ahead.

#5 - 01/08/2015 07:36 AM - Eugenie Lyzenko

The update evl_upd20150107a.zip has been committed in bzr as 10698.

#6 - 01/08/2015 08:45 AM - Greg Shah

Is there any behavior yet to be implemented?

#7 - 01/08/2015 08:55 AM - Eugenie Lyzenko

Greg Shah wrote:

> Is there any behavior yet to be implemented?

If we think about images processing here - handle image phrase in DEFINE BUTTON statement for size, stretching and options for:
LOAD-IMAGE* (filename[, x-offset , y-offset , width , height]) related to offset and size.

I planned to return to this during working on image phrase.

#8 - 01/21/2015 12:56 PM - Eugenie Lyzenko

The button widget image processing note. There are two places when we need to set/load image integration into button widget. First and simple one is using the LOAD-IMAGE*() methods. These methods set the image to button widgets already existed on the client side. And currently I have the proper solution for this case and it works fine.

The other place is handling the DEFINE BUTTON statement. The statement converts to the respective logical setImage*() methods on the server side. The main issue here is at the moment of server call setImage*() we do not have the button widget counterpart on the client side(even button constructor is not called). So we have nothing widget to set the image.

For now I see two possible solutions here.
1. Postpone the image loading to the moment when the button widget is created on the client side. This will require additional call from client to server to ask the image data. And as far as I understand this is not acceptable for performance reasons, right?
2. We create some storage on the client side to "cache" the images to apply when the client side widgets become ready to accept the images. This save from additional client-server conversation. This can be the static storage inside the ImageGuiImpl and will handle both button and image widgets. When the image transferred from this storage to real widget - the cached image can be removed to save some memory on the client side(images are memory hungry). As the image identifiers we can use widget ID and type(Up, Down, Insensitive or image widget owner) because the same image files can be used for all pictures to load.

So I'm inclining to approach #2. What do you think?

#9 - 01/21/2015 01:15 PM - Greg Shah

> Postpone the image loading to the moment when the button widget is created on the client side. This will require additional call from client to server to ask the image data. And as far as I understand this is not acceptable for performance reasons, right?

Actually, it won't be an additional call because even in approach 2 there is a call to the client for every image, right? This approach 1 is really just a deferral of the call to the time when the button is created. I'm OK with this.

Constantin: is there a better way to implement the loading as part of the button creation/push to the client?

#10 - 01/21/2015 01:25 PM - Constantin Asofiei

Greg Shah wrote:

> > Postpone the image loading to the moment when the button widget is created on the client side. This will require additional call from client to server to ask the image data. And as far as I understand this is not acceptable for performance reasons, right?
>
> Actually, it won't be an additional call because even in approach 2 there is a call to the client for every image, right? This approach 1 is really just a deferral of the call to the time when the button is created. I'm OK with this.
>
> Constantin: is there a better way to implement the loading as part of the button creation/push to the client?

If the image is loaded by 4GL right when the frame is defined, then is OK to load it and push it when the frame definition is pushed to the client. But if the image is loaded only when the frame (or button?) is realised, unless I've missed some discussion on the other image-related task, I don't think we know at this point the exact time when the image is pushed to the client, until we check this case: what happens if the PROPATH changes between the DEFINE BUTTON and the frame definition/view, so that now on the PROPATH there is another image file with the same name, but different contents (assuming relative paths are used)?

If image is really resolved and loaded at the frame definition, then I would just save it at the ButtonWidget on the server-side and ensure to push it to the client-side when the frame is realised. But can the button be part of more than one frame? can it show two images - if PROPATH changes - in different frames?

#11 - 01/21/2015 01:26 PM - Eugenie Lyzenko

> > Postpone the image loading to the moment when the button widget is created on the client side. This will require additional call from client to server to ask the image data. And as far as I understand this is not acceptable for performance reasons, right?
>
> Actually, it won't be an additional call because even in approach 2 there is a call to the client for every image, right?

Yes, but in approach 1 we have:
- try to load image, this is server->client call, if no widget at this time, return false
- ask the server for postponed image to load, client->server.
So in worse case we have 2 server<->client calls for the approach 1.

#12 - 01/21/2015 01:57 PM - Greg Shah

> But if the image is loaded only when the frame (or button?) is realised, unless I've missed some discussion on the other image-related task, I don't think we know at this point the exact time when the image is pushed to the client, until we check this case: what happens if the PROPATH changes between the DEFINE BUTTON and the frame definition/view, so that now on the PROPATH there is another image file with the same name, but different contents (assuming relative paths are used)?

Eugenie: please determine the answer to this question.

#13 - 01/21/2015 05:32 PM - Eugenie Lyzenko

> Eugenie: please determine the answer to this question.

The answer is the moment when button is realized is the point when 4GL loads image from currently set PROPATH directories. So if we change the PROPATH to something that contains the file with same name but new content - the image will be constructed from new content.

#14 - 01/21/2015 06:06 PM - Greg Shah

OK, please take this into account in your design.

1. I think that the loadImage() methods should still be used by the converted business logic, even for this DEFINE BUTTON case. The reason is simple: before realization I expect that the loadImage() methods will also have their effect deferred. In other words, this is probably a common feature that must be handled by loadImage().

2. If the button is already realized, then the processing occurs just as it is coded today.

3. If the button is not realized, then the load is deferred until realization.

#15 - 01/22/2015 06:15 PM - Eugenie Lyzenko

- File evl_upd20150122a.zip added

This update for review contains prototype for deferred image loading in button widget. Some notes:
1. There is only one server->client call, when ButtonWidget loads image(and client side is not yet ready)
2. The postponed image loading is happening when the Button calls initialize() method.
3. If image was taken from application jar file - the client stores the binary array data.
4. If image must be loaded from client file system - the client stores the file name as was requested on the server side.

5. If PROPATH is changing between DEFINE BUTTON and button realization - the file will be loaded from new PROPATH directories.

So in this implementation we keep the same server->client conversation as when handling LOAD-IMAGE*() methods.

Continue working with image widget itself for DEFINE IMAGE statement.

#16 - 01/23/2015 10:05 AM - Eugenie Lyzenko

- File evl_upd20150123a.zip added

This update for review adds DEFINE IMAGE handling for deferred data loading for image widget.

Continue working.

#17 - 01/23/2015 07:56 PM - Eugenie Lyzenko

- File evl_upd20150123b.zip added

This updates has code merged with recent code base.

One interesting thing I've found. If the method LOAD-IMAGE() is calling before the button or image widget associated with frame it can not be converted properly. For example:

```
...
def button b_sam label "Sample".

b_sam:load-image("file1").
...
```

converted to(and this is certainly wrong):

```
...
"file1".loadImage();
...
```

This is not happening if the source is:

```
...
def button b_sam label "Sample".

def frame fr
  b_sam
  with centered size 80 by 20.

b_sam:load-image("file1").
...
```

This properly converts to:

```
...
   frFrame.widgetBSam().loadImage("file1");
...
```

In general the currently suggested approach properly handles deferred and usual image loading for button/image widgets.

#18 - 01/26/2015 08:29 PM - Eugenie Lyzenko

Had to modify CommonWidget/GenericWidget classes to add loadImage*() methods support to handle dynamically allocated image/button widgets case.

#19 - 01/27/2015 08:27 PM - Eugenie Lyzenko

- File evl_upd20150127a.zip added

Small update to add method interfaces to support dynamic button/image widgets.

#20 - 01/28/2015 11:01 AM - Greg Shah

Code Review evl_upd20150127a.zip

I like it.

1. There are no real changes in ButtonWidget, just a history entry. I think that can be removed safely. :)

2. Please remember that we have a new policy for methods/attributes which are only available on a small number of widgets. I don't want the image loading methods to be added to CommonWidget, since only 2 widgets support these. Please create a separate interface (ImageSupport?) and implement that in ImageWidget and ButtonWidget. You will have to edit handle.java (unwrap method) and HandleCommon.java to add the new interface there.

3. Please do fix the problem in note 17.

4. What part of the changes were there for DEFINE IMAGE? I don't see any conversion changes.

#21 - 01/28/2015 11:24 AM - Eugenie Lyzenko

> 1. There are no real changes in ButtonWidget, just a history entry. I think that can be removed safely. :)

OK. Done.

> 4. What part of the changes were there for DEFINE IMAGE? I don't see any conversion changes.

The rule to properly convert DEFINE IMAGE was included in my previous committed update: evl_upd2015019c. The runtime support is in setImage()/loadImage() for image widget classes. Is it answering to your question?

#22 - 01/28/2015 12:13 PM - Greg Shah

- Status changed from New to WIP

- Target version set to Milestone 12

> Is it answering to your question?

Yes.

This update seems pretty close to something that can go into regression testing. Is there anything else needed except from the debugging you were going to do to confirm proper functioning?

#23 - 01/28/2015 12:25 PM - Eugenie Lyzenko

> This update seems pretty close to something that can go into regression testing. Is there anything else needed except from the debugging you were going to do to confirm proper functioning?

I'm thinking about note 2 above. I agree, we need separate interfaces for something that support by several widgets only. Because current approach looks a bit not effective. Even more we actually need two more classes, one for image and button(for LOAD-IMAGE) and other - only for button(the rest LOAD-IMAGE*() methods).

So we can run testing now or postpone to the time the interface rework is ready. What do you think?

BTW, the dynamically loaded images works OK. Need to test dynamic frames as well.

#24 - 01/28/2015 12:56 PM - Greg Shah

> So we can run testing now or postpone to the time the interface rework is ready. What do you think?

Do the rework now. When it is ready, post it. After a final code review, it will be into testing.

#25 - 01/28/2015 02:50 PM - Eugenie Lyzenko

- File evl_upd20150128a.zip added

Reworked image loading support.
- New interfaces added, reworked conversion rule for methods.
- bit change for image coordinates fixup.

The button/images loading works OK for dynamic widgets in dynamic frames too. So if update is OK we can start regression testing.

#26 - 01/28/2015 02:56 PM - Greg Shah

Code Review evl_upd20150128a.zip

It looks good. Please do get it tested.

#27 - 01/28/2015 03:01 PM - Eugenie Lyzenko

> It looks good. Please do get it tested.

OK. Starting the conversion testing stage(we have the rule changed).

#28 - 01/28/2015 07:02 PM - Eugenie Lyzenko

Conversion testing completed. The codes are identical. Starting the runtime testing.

#29 - 01/30/2015 07:56 AM - Eugenie Lyzenko

Finally I have got the main part completed without regression. Starting the CTRL-C tests.

#30 - 01/30/2015 11:43 AM - Eugenie Lyzenko

The CTRL-C part completed without regression. 3-way tests was started in a separate session. The results file on shared directory is: 10725_5c96d7c_20150130_evl.zip.

And because the evl_upd20150128a.zip does not cross other recent updates made since testing was started I think it is safe to check in and distribute. Is it OK?

#31 - 01/30/2015 04:57 PM - Greg Shah

Normally, the answer would be yes. But right now we are in the middle of a number of conflicting updates that have ThinClient changes. Those need to be checked in first. I'll let you know when we can "unfreeze" this one.

#32 - 01/30/2015 05:01 PM - Eugenie Lyzenko

> Normally, the answer would be yes. But right now we are in the middle of a number of conflicting updates that have ThinClient changes. Those need to be checked in first. I'll let you know when we can "unfreeze" this one.

OK. Not a problem.

#33 - 01/31/2015 05:36 PM - Eugenie Lyzenko

- File evl_upd20150131a.zip added

Merged with recent code base. Is it time to start new round or there will be more updates?

#34 - 01/31/2015 11:02 PM - Greg Shah

Sorry, there are 2 updates in front of you right now. Hynek's is first and then Constantin has one. Thanks for your patience.

#35 - 02/01/2015 03:52 PM - Eugenie Lyzenko

- File evl_upd20150201a.zip added

Update merged with the 10736 code base.

#36 - 02/02/2015 09:15 AM - Eugenie Lyzenko

- File evl_upd20150202a.zip added

Update merged with 10738 code base.

#37 - 02/02/2015 10:25 AM - Eugenie Lyzenko

The evl_upd20150202a.zip also merged with 10739.

#38 - 02/04/2015 11:16 AM - Eugenie Lyzenko

- File evl_upd20150204a.zip added

Merged with 10741 code base.

#39 - 02/04/2015 01:46 PM - Greg Shah

Please go ahead and do one more round of testing, using evl_upd20150204a.zip. If it passes, you can check it in.

#40 - 02/04/2015 02:20 PM - Eugenie Lyzenko

> Please go ahead and do one more round of testing, using evl_upd20150204a.zip. If it passes, you can check it in.

OK. The conversion testing is in progress.

#41 - 02/04/2015 05:09 PM - Eugenie Lyzenko

The conversion testing completed. The generated code is identical. Continue with runtime testing.

#42 - 02/05/2015 10:36 AM - Eugenie Lyzenko

The runtime testing almost completed. The only issue is the GSO 269 test. It failed with two main cycles but passed as standalone. I guess it is OK. But to have more sure I'm going to start one more main round cycle because the failure is pretty strange - INVALID FACILITY CODE for H1 which is known as valid facility.

#43 - 02/05/2015 10:53 AM - Greg Shah

I appreciate that you are taking extra care. Thanks.

#44 - 02/05/2015 04:08 PM - Eugenie Lyzenko

The testing completed. Results: 10742_5c96d7c_20150205_evl.zip. No regressions confirmed.

I've stopped the servers and going to check in and distribute the update.

#45 - 02/05/2015 04:15 PM - Eugenie Lyzenko

The update evl_upd20150204a.zip has been committed in bzr as 10744.

#46 - 02/25/2015 09:08 AM - Greg Shah

- Status changed from WIP to Closed

#47 - 11/16/2016 12:13 PM - Greg Shah

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

Files

evl_upd20150122a.zip	130 KB	01/22/2015	Eugenie Lyzenko
evl_upd20150123a.zip	134 KB	01/23/2015	Eugenie Lyzenko
evl_upd20150123b.zip	135 KB	01/24/2015	Eugenie Lyzenko
evl_upd20150127a.zip	157 KB	01/28/2015	Eugenie Lyzenko
evl_upd20150128a.zip	171 KB	01/28/2015	Eugenie Lyzenko
evl_upd20150131a.zip	171 KB	01/31/2015	Eugenie Lyzenko
evl_upd20150201a.zip	171 KB	02/01/2015	Eugenie Lyzenko
evl_upd20150202a.zip	171 KB	02/02/2015	Eugenie Lyzenko
evl_upd20150204a.zip	171 KB	02/04/2015	Eugenie Lyzenko