User Interface - Feature #4394

add some frame options and attributes

11/08/2019 04:23 PM - Greg Shah

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

Description

Related issues:

Related to User Interface - Feature #4164: implement misc UI features (frame ...	**WIP**
Related to User Interface - Bug #4627: CAN-QUERY incorrectly determines SCREE...	**New**

History

#1 - 11/08/2019 04:26 PM - Greg Shah

Add support for:

- FRAME-NAME attribute (this is actually a generic widget attribute but it will need to be implemented in the frame itself, possibly with conversion time changes needed)
- SCROLL frame option
- NO-ATTR-SPACE frame option

#2 - 03/24/2020 12:38 PM - Greg Shah

- Assignee set to Roger Borrello

#3 - 03/31/2020 10:59 AM - Greg Shah

- Related to Feature #4164: implement misc UI features (frame options, misc attributes) added

#4 - 03/31/2020 11:00 AM - Greg Shah

SCROLL and NO-ATTR-SPACE will be finished in #4164. The only thing needed for this task is FRAME-NAME.

#5 - 04/08/2020 11:03 AM - Roger Borrello

I cannot find any useful documentation related to how FRAME-NAME would be used in a testcase. Searching through our testcases, I only see that utilized in &Scoped-define FRAME-NAME ... statements.

Any simple example I could see?

#6 - 04/08/2020 11:29 AM - Greg Shah

The frame-name is just a string, right? So you can do anything with it to see the value (e.g. message widget:frame-name. or my-char-var = widget:frame-name..

Of course, you need to check it in multiple scenarios (chui/gui, before/after realization, on the frame itself/on contained widgets, with the unnamed/default frame, with an explicitly named frame).

#7 - 04/13/2020 04:07 PM - Roger Borrello

Greg Shah wrote:

> Add support for:
>
> - FRAME-NAME attribute (this is actually a generic widget attribute but it will need to be implemented in the frame itself, possibly with conversion time changes needed)

I'm researching, and there is already a keyword for FRAME-NAME which is kw_fr_name. However, there is no methods_attributes.rules support. My assumptions is that there will need to be getter/setter support added in FrameInterface.java and FrameWidget.java then conversion of the attribute to invoke. Then there's the actual frame name itself, but I have a hard time believing there isn't something already implemented to hold the frame name at this point... but I could be wrong.

#8 - 04/13/2020 04:19 PM - Greg Shah

> My assumptions is that there will need to be getter/setter support added in FrameInterface.java and FrameWidget.java then conversion of the attribute to invoke.

No, as noted above: "this is actually a generic widget attribute". Look at the 4GL docs. This can be called on any widget. So it should not be in the frame, but in CommonWidget/GenericWidget.

> Then there's the actual frame name itself, but I have a hard time believing there isn't something already implemented to hold the frame name at this point... but I could be wrong.

Search GenericFrame for frameName. You can obtain it by calling _getName(). All widgets know their frame. See GenericWidget.getFrame(). It may be null in some cases. This should be enough info to implement.

#9 - 04/13/2020 05:28 PM - Roger Borrello

Code ready for review. Additional testcases being developed. Tested via ON MOUSE-SELECT-CLICK of various widgets on a large frame, as well as on a separate default frame.

```
Committing to: /home/rfb/secure/code/p2j_repo/p2j/active/4231b/

modified rules/convert/methods_attributes.rules
modified src/com/goldencode/p2j/ui/CommonWidget.java
modified src/com/goldencode/p2j/ui/GenericWidget.java
Committed revision 11446.
```

#10 - 04/14/2020 10:21 AM - Roger Borrello

Current testcase has 2 of the same use of the :frame-name attribute, with very different conversions:

1. assign b4-frname = i:frame-name. converts to b4Frname.assign()
2. display i:frame-name. converts to f0Frame.widgeti().getFrameName()

The first doesn't compile. The second is what I was expecting.

#11 - 04/14/2020 10:30 AM - Greg Shah

The b4Frname.assign() is missing the f0Frame.widgeti().getFrameName() argument.

You'll have to fix it. Most likely, there is a missing frame-id annotation which causes the attribute to bypass being emitted.

#12 - 04/14/2020 10:37 AM - Greg Shah

Code Review Task Branch 4231b Revision 11446

The only issue is the use of new String() in GenericWidget.getFrameName(). That is unnecessary and is not ever done in Java. The correct code is return frame != null ? frame._getName() : "";.

What have your testcases shown about the behavior? If the frame is not set for a widget (e.g. a CREATE <widget> which has no FRAME attribute ever set), then does it really return empty string? I would have guessed it to return unknown value. Anyway, please document the correct behavior so I can evaluate your work.

#13 - 04/14/2020 12:26 PM - Roger Borrello

Greg Shah wrote:

> The b4Frname.assign() is missing the f0Frame.widgeti().getFrameName() argument.
>
> You'll have to fix it. Most likely, there is a missing frame-id annotation which causes the attribute to bypass being emitted.

Looking at more code emitted... would this be similar to why this code:

```
  b4col = frame-col(f0)
  b4row = frame-row(f0)
```

converts to:

```
b4col.assign(0);
b4row.assign(0);
```

Those don't look correct.

#14 - 04/14/2020 12:53 PM - Roger Borrello

Roger Borrello wrote:

> Looking at more code emitted... would this be similar to why this code:
> [...]
> converts to:
> [...]
>
> Those don't look correct.

It looks like frame_scoping.rules has a check for this functions which reference the frame:

```
      <!-- process functions which reference the frame -->
      <rule>
         evalLib("is_func", type)               and
         this.getAnnotation("oldtype") != null  and
         (getNoteLong("oldtype") == prog.kw_fr_row  or
          getNoteLong("oldtype") == prog.kw_fr_col  or
          getNoteLong("oldtype") == prog.kw_fr_line or
          getNoteLong("oldtype") == prog.kw_fr_down)

         <rule>
            this.getFirstChild() != null and
            this.getFirstChild().type == prog.wid_frame

            <!-- named frame as a parameter -->
            <action on="true">
               execLib("add_node_ex",
                       execLib("new_frame", this.getFirstChild().text, false), copy)
            </action>

            <!-- no valid named frame, but check if there is a parameter -->
            <rule on="false">this.getFirstChild() == null

               <!-- no parameter means this is an implicit reference to the default frame -->
               <action>execLib("add_node", copy)</action>

               <!-- invalid (non-existing) frame name case, all of these funcs always return 0
                    in this case; we will hard code it here because there is no frame reference
                    to emit -->
               <action on="false">copy.type = prog.num_literal</action>
               <action on="false">copy.text = "0"</action>

            </rule>
         </rule>
      </rule>
```

Do I need to look at these, and make a more in-depth analysis of whether or not there should be a frame reference?

#15 - 04/14/2020 02:06 PM - Greg Shah

We can't go on an 2 week effort to dig into this. It would be good to fix these as well. The issue is probably different.

In the function case (as opposed to your attribute case), the expression this.getFirstChild() != null and this.getFirstChild().type == prog.wid_frame fails. This is the reason for the issue. f0 should appear as a wid_frame in the tree. Fix that and these will work.

#16 - 04/14/2020 02:07 PM - Roger Borrello

Greg Shah wrote:

> The b4Frname.assign() is missing the f0Frame.widgeti().getFrameName() argument.
>
> You'll have to fix it. Most likely, there is a missing frame-id annotation which causes the attribute to bypass being emitted.

In this case, I am referencing a widget that is not associated with any frame. I'll look at 4GL to see what happens.

#17 - 04/14/2020 02:35 PM - Roger Borrello

Roger Borrello wrote:

> Greg Shah wrote:
>
> > The b4Frname.assign() is missing the f0Frame.widgeti().getFrameName() argument.
> >
> > You'll have to fix it. Most likely, there is a missing frame-id annotation which causes the attribute to bypass being emitted.
>
> In this case, I am referencing a widget that is not associated with any frame. I'll look at 4GL to see what happens.

```
Could not find i in a frame. Try qualifying the reference with an IN FRAME phrase. (3566)
** z:\testcases\uast\frame_layout\frame_widget_attributes_helper.i Could not understand line 2. (196)
```

There should be some sort of error or warning during conversion, I would think, rather than letting this get to the compilation stage.

#18 - 04/14/2020 02:38 PM - Greg Shah

As someone wise once said: "The problem was a loose nut behind the wheel.". :)

Our current implementation assumes you are providing correct syntax.

#19 - 04/15/2020 09:46 AM - Roger Borrello

Greg Shah wrote:

> Code Review Task Branch 4231b Revision 11446
>
> The only issue is the use of new String() in GenericWidget.getFrameName(). That is unnecessary and is not ever done in Java. The correct code is return frame != null ? frame._getName() : "";.
>
> What have your testcases shown about the behavior? If the frame is not set for a widget (e.g. a CREATE <widget> which has no FRAME attribute ever set), then does it really return empty string? I would have guessed it to return unknown value. Anyway, please document the correct behavior so I can evaluate your work.

When you look for frame-name on a widget not yet on a frame, it shows "?" in 4GL.

f0: b4-col/b4-row= 0 / 0 aft-col/aft-row= 1 / 1 col/row= 1 / 1 h1: b4-frname/aft-frname= ? / ? x/y= 0 / 0 fx/fy= ? / ? c/r= 1 / 1 fc/fr= ? / ?

Legend:

```
   "{&fr}: b4-col/b4-row="b4-col "/" b4-row 
   "aft-col/aft-row="frame-col({&fr}) "/" frame-row({&fr}) 
   "col/row="{&colval} "/" {&rowval}
   "{&wid}:" 
   "b4-frname/aft-frname="b4-frname "/" {&wid}:frame-name
   "x/y=" {&wid}:x "/" {&wid}:y                   /* pix loc of wid left/top edge rel to left parent edge */
   "fx/fy=" {&wid}:frame-x "/" {&wid}:frame-y     /* pix loc of wid left/top edge rel to ul of frame */
   "c/r=" {&wid}:column "/" {&wid}:row            /* col/row pos of wid left/top edge rel to left/top (same as
 x/y) */
   "fc/fr=" {&wid}:frame-col "/" {&wid}:frame-row /* Dec col pos of wid left/top edge rel to ul of wid frame *
/
```

#20 - 04/15/2020 09:53 AM - Greg Shah

Right. So your existing code is not correct. I suspect that this is not the string "?" but it is just how the language statement prints unknown value. Check it.

#21 - 04/15/2020 10:22 AM - Roger Borrello

Greg Shah wrote:

> Right. So your existing code is not correct. I suspect that this is not the string "?" but it is just how the language statement prints unknown value. Check it.

There is a name() method in GenericWidget that returns the name of the widget as character. When I change getFrameName to return character, I get the desired unknown value:
f0: b4-col/b4-row= 1 / 1 aft-col/aft-row= 1 / 1 col/row= 1 / 1 h1: b4-frname/aft-frname= ? / ? x/y= 0 / 0 fx/fy= 0 / 0 c/r= 1 / 1 fc/fr= 1 / 1

There are other widget attributes that aren't matching up (frame-x, frame-y, frame-col, and frame-row), so I am looking those over.

The new code in getFrameName:

```
   public character getFrameName()
   {
      return frame != null ? new character(frame._getName()) : new character();
   }
```

#22 - 04/15/2020 10:38 AM - Greg Shah

We always must return BDT values from attribute getters. We never return String instances.

#23 - 04/15/2020 11:38 AM - Roger Borrello

Initial attribute values being returned for frame-col and frame-row are incorrect. They are coming from the client, and I could use a hint how to debug that area of the client. Like where to set a breakpoint.

#24 - 04/15/2020 12:11 PM - Roger Borrello

Roger Borrello wrote:

> Initial attribute values being returned for frame-col and frame-row are incorrect. They are coming from the client, and I could use a hint how to debug that area of the client. Like where to set a breakpoint.

I see that I need to include suspend. Nevermind.

#25 - 04/15/2020 02:21 PM - Roger Borrello

By updating getFrameRow and getFrameCol in ThinClient.java to return a 0, if the frame is not visible, I get matching results to the 4GL.

```
   public double getFrameRow(int frameId)
   {
      Frame frame = getFrame(frameId);
      double result = 0; // Changed from 1 to 0

      if (frame != null && frame.isVisible())
      {
         result = frame.location().y + 1;
      }

      return result;
   }

   {
      Frame frame = getFrame(frameId);
      double result = -1; // Changed from 0 to -1

      if (frame != null && frame.isVisible() && !frame.isRedirected())
      {
         result = ConfigHelper.getAssignedColumn(frame.config());
      }

      if (result == BaseConfig.INV_COORD)
      {
         result = 0;
      }

      return ++result;
   }
```

I am not particularly fond of the way in which they do not proceed in the same manner with returning an incremented value, versus starting with the same initialization. I want to change it, but Greg mentioned there are some complex computations that take place with respect to coordinates. So I thought I'd ask if these look feasible, or if it's ok to re-swizzle the code a little to make them more consistent to each other.

#26 - 04/16/2020 06:36 PM - Hynek Cihlar

Roger Borrello wrote:

> By updating getFrameRow and getFrameCol in ThinClient.java to return a 0, if the frame is not visible, I get matching results to the 4GL.

Code review, the changes are OK.

#27 - 04/17/2020 06:39 PM - Roger Borrello

- Status changed from New to WIP

- % Done changed from 0 to 90

#28 - 04/22/2020 06:00 PM - Roger Borrello

Getting back to FRAME-NAME, it looks like convert/frame_construction.rules retrieves the critical data using:

```
         <action>javaname = getNoteString("javaname")</action>
         <action>frameName = getNoteString("name")</action>
```

The frameName ends up being all lowercase, which is not the same as the 4GL outputs for FRAME-NAME. Shouldn't this be javaname for the frame, in order to preserve the case?

#29 - 04/22/2020 06:07 PM - Greg Shah

No. Nein. Nyet. Non. Nee.

#30 - 04/22/2020 06:08 PM - Roger Borrello

Greg Shah wrote:

> No. Nein. Nyet. Non. Nee.

Tried updating that, and it looks like it should be using name. However, the annotation is not correct.

#31 - 04/22/2020 06:09 PM - Greg Shah

OK, perhaps I need to clarify.

javaname is a converted Java name and will have been transformed to remove all the gorp that can be in a 4GL name but that cannot appear in Java. Never consider this to be a legacy name.

And you do need the legacy name, right?

So: save off the legacy name before it gets lowercased. Store than in an annotation and stuff it into the frame name.

#32 - 04/22/2020 06:18 PM - Roger Borrello

Greg Shah wrote:

> OK, perhaps I need to clarify.
>
> javaname is a converted Java name and will have been transformed to remove all the gorp that can be in a 4GL name but that cannot appear in Java. Never consider this to be a legacy name.
>
> And you do need the legacy name, right?
>
> So: save off the legacy name before it gets lowercased. Store than in an annotation and stuff it into the frame name.

I understand about javaname. I could either add another annotation in the frame_alloc node, or make the name annotation the correct case. I'll have to see if that's used elsewhere.

#33 - 04/23/2020 11:14 AM - Roger Borrello

About a year ago, Constantin made a specific update to annotations/frame_scoping.rules:

```
CA  20190327          Fixed dictionary for frame names (values must be lowercased).
```

```
      <!-- set name to the current frame scope -->
      <function name="name_scope">
         <parameter name="fname" type="java.lang.String" />
         <variable name="var"    type="java.util.Map" />
         <variable name="local"  type="java.util.Map" />

         <rule>fname != null and !fname.equals('')
            <action>var = execLib("get_named_frame", fname)</action>

            <rule>var != null
               <action>local = #(java.util.Map)
                       lookupDictionaryObject(currScope, frameVar)</action>
               <action>evalLib("merge_scopes", local, var)</action>
            </rule>
            <rule>var == null
               <!-- no such frame -->
               <action>var = #(java.util.Map)
                       lookupDictionaryObject(currScope, frameVar)</action>
               <action>var.put(frameName, fname)</action>

               <action>execLib("add_frame", fname, var)</action>
            </rule>
            <action>addDictionaryObject(currScope, frameVar, var)</action>
            <action>addDictionaryObject(currScope, frameName, fname.toLowerCase())</action>
         </rule>
      </function>
```

Unfortunately it is difficult to discern the backstory, because this has downstream effects in convert/frame_construction.rules where the name is pulled when frame_alloc is found.

I am still looking if I can add "legacy_name" annotation, but it's a challenge to work in the mapping methods of frame_scoping.rules.

#34 - 04/23/2020 11:19 AM - Greg Shah

Why mess around? Just store it before frame scoping occurs.

#35 - 04/23/2020 11:24 AM - Roger Borrello

Greg Shah wrote:

> Why mess around? Just store it before frame scoping occurs.

From what I know, it's pay-me-now, or pay-me-later, because frame_scoping is where the FRAME_ALLOC node is created, and that's where the frame is constructed in conversion. If I put the annotation in the frame definition, I'll have to find it during the construction phase. That could very well be the easier path.

#36 - 04/23/2020 06:54 PM - Roger Borrello

Made changes in frame_scoping.rules to create a "parallel" legacy name that is handled in the same manner as the frame name, except the original case is maintained throughout. At the end, when the frame_alloc is created, a new annotation legacy_name is stored in the AST. This is used by frame_construction.rules to build the createFrame method.

Review needed, and other testcases related to shared frames and browses will need to be performed.

```
Committing to: /home/rfb/secure/code/p2j_repo/p2j/active/4231b/

modified rules/annotations/frame_scoping.rules
modified rules/convert/frame_construction.rules
Committed revision 11483.
```

#37 - 04/24/2020 09:57 AM - Roger Borrello

I was thinking of the ramifications of using the actual frame name as specified in the 4GL, instead of the lowercase name. I give 2 alternatives, and then a discertation about the current method below.

1. Add the legacy_name as a 3rd parameter to GenericFrame.createFrame instead of replacing the 2nd, or
2. Create a setFrameName setter to add to the frame definition

```
FrameLocAttributesGet3Pbframe pbframeFrame = GenericFrame.createFrame(FrameLocAttributesGet3Pbframe.class, "pb
Frame");
```

I'm not sure which is better. I do know the concept of the FRAME-NAME may have 2 definitions, one for the implementation internally, and one for the exposed 4GL interface. If the use of the mixed-case original 4GL name is OK to use throughout the implementation, I just need to complete this by storing it as "name" in the annotations, rather than creating a new "legacy_name" and letting all the other code handle it as-is. It would just be the original name, rather than a manufactured name.

#38 - 04/24/2020 10:05 AM - Greg Shah

I doubt that is a good idea. The original 4GL code allows specifying the name in any case when you refer to the frame. So it can be 100 different versions throughout a file. I suspect this is why Constantin lowercased name recently, so that all of the TRPL code would process using the same name.

And that is fine, except for when the legacy name is accessed as a character attribute or through a 4GL built-in function. Then we need to report the same thing as the 4GL.

Which brings up a point which I should have mentioned previously. Make sure you test these scenarios:

- DEFINE FRAME and different names used everywhere
- FORM and different names used everywhere
- no DEFINE FRAME or FORM and different names used everywhere

By "everywhere", I mean all the other frame phrase references:

```
DEFINE FRAME mY-fRaMe ...
DISPLAY ... WITH FRAME My-Frame.
DOWN 1 WITH FRAME MY-framE.
```

Capice?

#39 - 04/24/2020 10:08 AM - Greg Shah

Hint: I wonder if the 4GL returns (from FRAME-NAME()) the first frame name that is defined, the last one that is defined or some other ridiculous random feverdream.

#40 - 04/24/2020 10:58 AM - Roger Borrello

Greg Shah wrote:

> Hint: I wonder if the 4GL returns (from FRAME-NAME()) the first frame name that is defined, the last one that is defined or some other ridiculous random feverdream.

Thankfully it utilizes the first mention of the frame.

```
DISPLAY i ch WITH FRAME fRame1.
FORM i ch WITH FRAME FramE1 2 DOWN.

DO  i = 1 TO 2:
   ch = STRING(i).
   DISPLAY i ch WITH FRAME fRame1.
   i:PRIVATE-DATA IN FRAME frAme1 = STRING(i).
   ch:PRIVATE-DATA IN FRAME fraMe1 = STRING(i).
   DOWN WITH FRAME framE1.
END.
MESSAGE "ch:FRAME-NAME=" ch:FRAME-NAME.
```

Displays the first mention, regardless of what I move at the top (DISPLAY or FORM).

#41 - 04/24/2020 11:48 AM - Greg Shah

Is FRAME-NAME() ready for review?

#42 - 04/24/2020 12:47 PM - Roger Borrello

Greg Shah wrote:

> Is FRAME-NAME() ready for review?

No... Is one of these a better choice than the other?

1. Add the legacy_name as a 3rd parameter to GenericFrame.createFrame instead of replacing the 2nd, or
2. Create a setFrameName setter to add to the frame definition

#43 - 04/24/2020 12:48 PM - Greg Shah

Today, what is that 2nd parameter used for other than as the legacy name?

#44 - 04/24/2020 01:16 PM - Roger Borrello

Greg Shah wrote:

> Today, what is that 2nd parameter used for other than as the legacy name?

Two main things...

1. The frame's name (this.frameName) which is actually supporting FRAME-NAME attribute.
2. The frame's name (this.name) which is built by concatenating this.frameName+##+this.config.Class.getName().

I suppose I could keep the 2nd parameter as the original name, and in the GenericFrame.coreInitialize method, perform a lowercase of it before building up the 2nd frame name, and all would be as it was.

#45 - 04/24/2020 01:35 PM - Greg Shah

> I suppose I could keep the 2nd parameter as the original name, and in the GenericFrame.coreInitialize method, perform a lowercase of it before building up the 2nd frame name, and all would be as it was.

I prefer that idea.

Constantin: Any objection?

#46 - 04/24/2020 01:51 PM - Constantin Asofiei

Greg Shah wrote:

> > I suppose I could keep the 2nd parameter as the original name, and in the GenericFrame.coreInitialize method, perform a lowercase of it before building up the 2nd frame name, and all would be as it was.
>
> I prefer that idea.
>
> Constantin: Any objection?

No objection, this approach should work.

#47 - 04/24/2020 03:40 PM - Roger Borrello

I have updated and my testcase passes.

```
Committing to: /home/rfb/secure/code/p2j_repo/p2j/active/4231b/

modified src/com/goldencode/p2j/ui/GenericFrame.java
Committed revision 11487.
```

However, there is a failure in this particular testcase (./uast/frame_layout/widget-hierarchy_fname.p) when it is executing @CAN-QUERY where it allows SCREEN-VALUE to pass, but in 4GL this shows as not a queryable attribute for FRAME widget. The testcase then attempt to read it, and ** SCREEN-VALUE is not a queryable attribute for FRAME widget. (4025) is displayed.

Will open a bug.

#48 - 04/24/2020 03:47 PM - Roger Borrello

- % Done changed from 90 to 100

#49 - 04/24/2020 03:48 PM - Roger Borrello

- Related to Bug #4627: CAN-QUERY incorrectly determines SCREEN-VALUE on a FRAME widget is queryable added

#50 - 04/25/2020 10:41 AM - Greg Shah

Code Review Task Branch 4231b Revisions 11483, 11486, 11487

The changes are good.

1. Please modify the GenericFrame javadoc so that it is clear what is the legacy 4GL name (case-preserved), what is the internal frame name (lowercased) and so forth. I think coreInitialize() needs edits in this regard as well as the name and frameName members.

2. In java_templates.tpl there is the construct_frame template. The javaname target is a misnomer. It now should be called legacy_name. Please change it there and you'll have to change it everywhere that template is used. This is just to make the code more clear, otherwise it will confuse those who follow you.

#51 - 04/27/2020 10:13 AM - Roger Borrello

Greg Shah wrote:

> Code Review Task Branch 4231b Revisions 11483, 11486, 11487

> The changes are good.
>
> 1. Please modify the GenericFrame javadoc so that it is clear what is the legacy 4GL name (case-preserved), what is the internal frame name (lowercased) and so forth. I think coreInitialize() needs edits in this regard as well as the name and frameName members.

frameName is easy to find and document. But name is inherited from someplace else. Would that be cfg.Configuration? Should I just document GenericFrame or go beyond?

#52 - 04/27/2020 10:22 AM - Roger Borrello

Greg Shah wrote:

> 2. In java_templates.tpl there is the construct_frame template. The javaname target is a misnomer. It now should be called legacy_name. Please change it there and you'll have to change it everywhere that template is used. This is just to make the code more clear, otherwise it will confuse those who follow you.

I only see it used in frame_construction.rules.

#53 - 04/27/2020 10:38 AM - Greg Shah

> Should I just document GenericFrame or go beyond?

Just GenericFrame is fine.

#54 - 04/27/2020 12:25 PM - Roger Borrello

Ready for test.

```
Committing to: /home/rfb/secure/code/p2j_repo/p2j/active/4231b/

modified rules/convert/frame_construction.rules
modified rules/convert/java_templates.tpl
modified src/com/goldencode/p2j/ui/GenericFrame.java
Committed revision 11488.
```

#55 - 04/27/2020 04:05 PM - Greg Shah

Code Review Task Branch 4231b Revision 11488

The only issue is a cut/paste problem in frame_construction.rules which should not have history entry 132. It is just a "sub-entry" of the already existing history entry 016.

#56 - 04/28/2020 09:49 AM - Greg Shah

- Status changed from WIP to Test

With the change in 4231b rev 11492, this is now complete.

#57 - 05/01/2020 01:51 PM - Roger Borrello

- % Done changed from 100 to 30

- Status changed from Test to WIP

The other 2 tasks are TBD, so backing up this task to 30%.

- SCROLL frame option
- NO-ATTR-SPACE frame option

#58 - 05/14/2020 06:33 PM - Roger Borrello

- % Done changed from 30 to 100

According to #4394-4 this task should only cover FRAME-NAME, so it can be moved to Test.

#59 - 05/15/2020 07:32 AM - Greg Shah

- Status changed from WIP to Test

#60 - 06/20/2020 12:00 PM - Greg Shah

- Status changed from Test to Closed

Task branch 4231b has been merged to trunk as revision 11347.