User Interface - Feature #3275

improve browse support

04/18/2017 12:27 PM - Greg Shah

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

Description

Related issues:

Related to User Interface - Bug #3306: Improve intermediate state in ROW-DISP...	**New**	**06/29/2017**
Related to User Interface - Bug #2799: browse widget doesn't execute validati...	**New**	
Related to User Interface - Feature #2628: Non-fill-in column support in browse.	**WIP**	**07/30/2015**
Related to User Interface - Bug #3335: buggy no-validate in browse widget	**Closed**	
Related to User Interface - Bug #3361: In ChUI browse does not start editing ...	**New**	**10/24/2017**

History

#1 - 04/18/2017 12:29 PM - Greg Shah

- Status changed from New to WIP

- Assignee set to Stanislav Lomany

Browse Attributes, Methods and Options to be added to FWD

Attributes

~~CURRENT-COLUMN~~
~~INDEX~~
~~COLUMN-READ-ONLY~~
~~ROW-HEIGHT-PIXELS~~
~~BUFFER-FIELD~~
NO-VALIDATE
NUM-ENTRIES

Methods

~~ADD-CALC-COLUMN~~ and ~~ROW-DISPLAY event processing~~
~~SET-SORT-ARROW~~
~~CLEAR-SORT-ARROWS~~
~~SCROLL-TO-CURRENT-ROW~~
~~SCROLL-TO-SELECTED-ROW~~

Options

~~LABEL-FGCOLOR~~
~~VIEW-AS-TOGGLE-BOX~~
~~LABEL-FONT (runtime only)~~
~~COLUMN-FGCOLOR~~
NO-AUTO-VALIDATE
~~NO-SCROLLBAR-VERTICAL (runtime only)~~
NO-VALIDATE (runtime only)
~~AUTO-RETURN (runtime only)~~
~~WIDTH-PIXELS (conversion only)~~
~~NO-EMPTY-SPACE~~
~~ROW-HEIGHT-PIXELS~~

I'm assuming that ALLOW-COLUMN-SEARCHING and SORT-ASCENDING are 100% complete in 2959b and thus do not need to be done here.

Correct?

#2 - 04/18/2017 02:21 PM - Stanislav Lomany

Correct.

#3 - 04/19/2017 07:53 AM - Stanislav Lomany

CURRENT-COLUMN is already implemented.

#4 - 04/19/2017 07:59 AM - Greg Shah

> CURRENT-COLUMN is already implemented.

Please update the gaps/expressions.rules regarding this and all the other changes you make in this task.

#5 - 04/27/2017 08:36 AM - Greg Shah

Can you please summarize the status of this task?

#6 - 04/27/2017 08:40 AM - Stanislav Lomany

INDEX and COLUMN-READ-ONLY are done, finishing ROW-HEIGHT-PIXELS.

#7 - 04/27/2017 08:53 AM - Greg Shah

What branch can I use to see the changes?

#8 - 04/27/2017 08:56 AM - Stanislav Lomany

2892a. ROW-HEIGHT-PIXELS is not checked in yet.

#9 - 04/27/2017 09:14 AM - Stanislav Lomany

Class design question.
BaseEntity contains two functions setSizePixels and setSizeChars which set width/height of a widget, but at first it rounds and trims values and ensures that character size corresponds a whole number of pixels.
There are two attributes which share this logic: BROWSE-COLUMN:WIDTH (browse column doesn't extend BaseEntity) and BROWSE:ROW-HEIGHT. But the logic in BaseEntity is bound to BaseEntity.set*Width/Height*Pixels/Chars*Worker methods.
I suggest to separate parameter preparation and parameter setting logic so the preparation part can be used for BROWSE-COLUMN:WIDTH and BROWSE:ROW-HEIGHT. What came to my mind is to create

```
public static Object[] BaseEntity.prepareSizePixels(Integer value, SizeAttribute attr, GenericWidget widget)
public static Object[] BaseEntity.prepareSizeChars(Double value, SizeAttribute attr, GenericWidget widget)
```

function which return a pair of (CHARS, PIXELS) values. What do you think?

#10 - 04/27/2017 09:18 AM - Greg Shah

> function which return a pair of (CHARS, PIXELS) values. What do you think?

I like the idea. The only change I would like to see is to create a small static inner class that is a simple container (no methods for get/set, direct package private access to the members) to return the pair of (CHARS, PIXELS) data. This adds type safety and is easier to read/maintain.

Hynek: please see #3275-9 and add your assessment.

#11 - 04/27/2017 09:20 AM - Greg Shah

Code Review 2892a Revision 11152

The changes are OK.

My only concern is that the getIndex() implementation has no support for variable extents. How does the 4GL handle that case?

#12 - 04/27/2017 09:49 AM - Hynek Cihlar

Stanislav Lomany wrote:

> Class design question.
> BaseEntity contains two functions setSizePixels and setSizeChars which set width/height of a widget, but at first it rounds and trims values and ensures that character size corresponds a whole number of pixels.
> There are two attributes which share this logic: BROWSE-COLUMN:WIDTH (browse column doesn't extend BaseEntity) and BROWSE:ROW-HEIGHT. But the logic in BaseEntity is bound to BaseEntity.set*Width/Height*Pixels/Chars*Worker methods.
> I suggest to separate parameter preparation and parameter setting logic so the preparation part can be used for BROWSE-COLUMN:WIDTH and BROWSE:ROW-HEIGHT. What came to my mind is to create
> [...]
> function which return a pair of (CHARS, PIXELS) values. What do you think?

This sounds OK, considering that there isn't much else to do to deduplicate the logic. Just please declare the prepare* methods with package default access level.

#13 - 04/27/2017 06:29 PM - Stanislav Lomany

> My only concern is that the getIndex() implementation has no support for variable extents. How does the 4GL handle that case?

It returns correct index for an extent variable. I'll add this as TODO because Element doesn't provide extent information and conversion changes are required.

#14 - 04/28/2017 08:24 AM - Greg Shah

> I'll add this as TODO because Element doesn't provide extent information and conversion changes are required.

The changes seem to be modest. I don't want to leave this as a partial item if we can reasonably encode the index value into the Element instance. Please take a look.

#15 - 05/03/2017 06:02 AM - Stanislav Lomany

> if we can reasonably encode the index value into the Element instance. Please take a look.

Currently extent variables in browse are converted like

```
new Element(subscript(ext, 2), frFrame.widgetBrExtArray1Column())
```

I suggest for DEFINE BROWSE case replace subscript with some accessor:

```
new Element(new IndexedAccessor(ext, 2), frFrame.widgetBrExtArray1Column())
```

then in BROWSE-COLUMN:INDEX use instanceof to detect it. Is it OK?

#16 - 05/03/2017 08:25 AM - Greg Shah

I like the approach. Go with it.

#17 - 05/04/2017 06:17 AM - Stanislav Lomany

NO-VALIDATE option and attribute and NO-AUTO-VALIDATE option all assume that schema-level validation works for static and dynamic browse. But it is not yet implemented. For fill-ins schema-level validation is implemented by assigning a validator at conversion stage:

```
frFrame.widgetBookId().setValidation(((ValidationExpr<integer>) (integer bookId_) -> isNotEqual(bookId_, 33)),
 "Book id 33!");
```

How should we implement schema-level validation for a dynamic browse where column fields are determined at runtime?

Also, what project uses these option? I want to take a look at real-life usage.

#18 - 05/04/2017 08:59 AM - Greg Shah

> How should we implement schema-level validation for a dynamic browse where column fields are determined at runtime?

I suspect we need to dynamically convert the VALEXP/VALMSG from the schema at runtime using a same or similar technique that we use for dynamic database.

The only real tricky part I see here is if these expressions can reference variables, widgets and other arbitrary state in the containing business logic. Static validation expressions can certainly do this, as can static database WHERE clauses. However, for dynamic database it was found that references to non-buffer business logic state were not possible. Please test this to see if it is needed.

> Also, what project uses these option? I want to take a look at real-life usage.

See #3257.

#19 - 05/04/2017 04:21 PM - Stanislav Lomany

FYI, in order to make ADD-CALC-COLUMN useful, I should also enable ROW-DISPLAY events for GUI browse and ensure that coloring/screen-values work properly.

#20 - 05/07/2017 10:40 AM - Stanislav Lomany

> The only real tricky part I see here is if these expressions can reference variables, widgets and other arbitrary state in the containing business logic. ... Please test this to see if it is needed.

As far as I tested, it cannot reference local variables. But it can reference temp tables. Also it can reference records from other databases, but it doesn't seem to work (no errors are produced).

#21 - 05/07/2017 10:42 AM - Stanislav Lomany

It can reference records from the same database (e.g. in CAN-FIND).

#22 - 05/07/2017 05:08 PM - Stanislav Lomany

> I suspect we need to dynamically convert the VALEXP/VALMSG from the schema at runtime using a same or similar technique that we use for dynamic database.

Do you mean that we should adapt DynamicQueryHelper.parse for that?

#23 - 05/08/2017 08:49 AM - Greg Shah

> it cannot reference local variables.

That is good news.

> it can reference temp tables
> It can reference records from the same database (e.g. in CAN-FIND).

OK, it will be important to understand the limits of this. For example, does the buffer (temp-table or within the same database) need to be part of the query which is bound to the browse to make this work? Or can any arbitrary temp-table or same database buffer be referenced?

> > I suspect we need to dynamically convert the VALEXP/VALMSG from the schema at runtime using a same or similar technique that we use for dynamic database.
>
> Do you mean that we should adapt DynamicQueryHelper.parse for that?

Yes, this is one way. Actually, we might do an alternate version that is designed for the VALEXP/VALMSG case. I don't want to break the current dynamic database code and it is possible that the changes would be pretty intrusive. On the other hand, if we can reuse code or make it common without breaking the current dynamic database implementation, it would be good.

I do think that the RuntimeJastInterpreter code can be enhanced and made into common code.

#24 - 05/19/2017 07:11 AM - Greg Shah

Another question about the dynamic validation expressions: can they reference user-defined functions in the business logic?

#25 - 05/19/2017 07:11 AM - Greg Shah

Please summarize the status of this task. Which items are fully complete? Which items are in process?

#26 - 05/19/2017 01:16 PM - Stanislav Lomany

Done:

```
CURRENT-COLUMN
INDEX
COLUMN-READ-ONLY
ROW-HEIGHT-PIXELS
BUFFER-FIELD
```

Was investigating validation behavior as a part of NO-VALIDATE, but at some point switched to ADD-CALC-COLUMN. "Calc column" is populated using SCREEN-VALUE. It wasn't supported (as well as other cell-specific color attributes), so I implemented it all and now working on calc column itself.

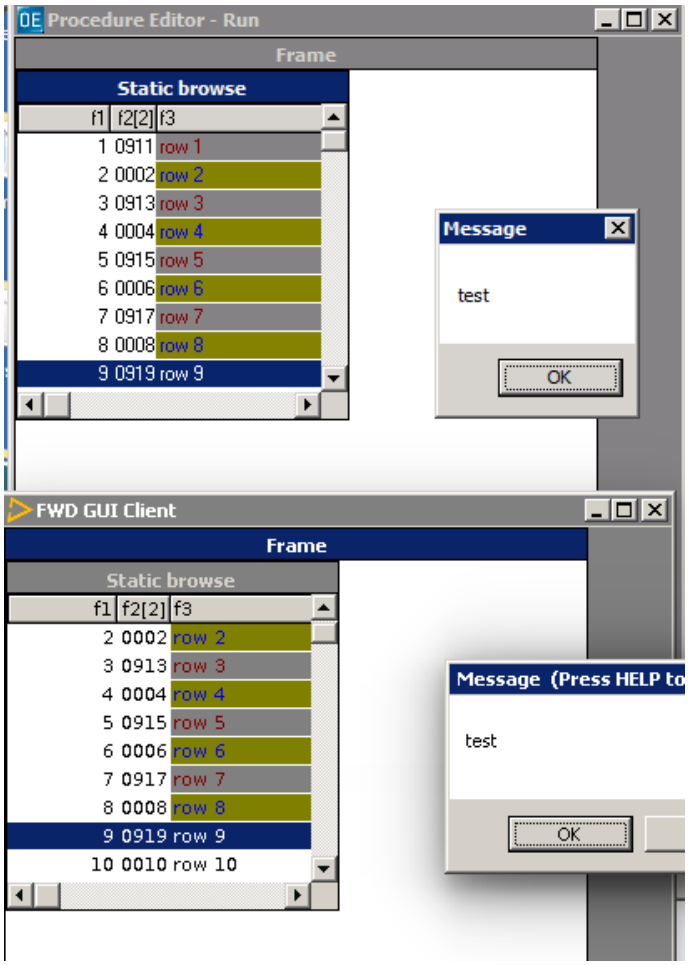
#27 - 05/24/2017 05:06 PM - Stanislav Lomany

I found that we have problems if a value cannot be displayed using column format. In ChUI error message is posted every time the cell it is drawn, which happens too often.
In GUI we have abend because an error message box is displayed in browse.draw(). We have the same abend if we try to display a message box in ROW-DISPLAY trigger.
I'm going to fire ROW-DISPLAY trigger and handle invalid format on browse data update.

#28 - 05/25/2017 10:02 AM - Stanislav Lomany

- File row-display.png added

I've removed ROW-DISPLAY triggering from draw() and noticed some flickering. Turned out that historically we do not raise ROW-DISPLAY in the proper place. That didn't matter because normally ROW-DISPLAY do not suppose user interactions. But this image shows that in P2J ROW-DISPLAY is triggered just *after* we added new row to cache, while in 4GL - just *before* we do that ("test" below is the message from ROW-DISPLAY trigger).



So the sequence

1. Get new rows.
2. Add new rows to cache.
3. Trigger row display for each new row.
4. Changes can be applied to a cached row in ROW-DISPLAY.
5. Repaint.

Should be changed to:

1. Get new rows.
2. Trigger row display for each new row.
3. Changes can be applied to the set of changes in ROW-DISPLAY.
4. Add new rows to cache.
5. Apply set of changes to cache.
6. Repaint.

Should I do that?

#29 - 05/25/2017 10:05 AM - Greg Shah

Should I do that?

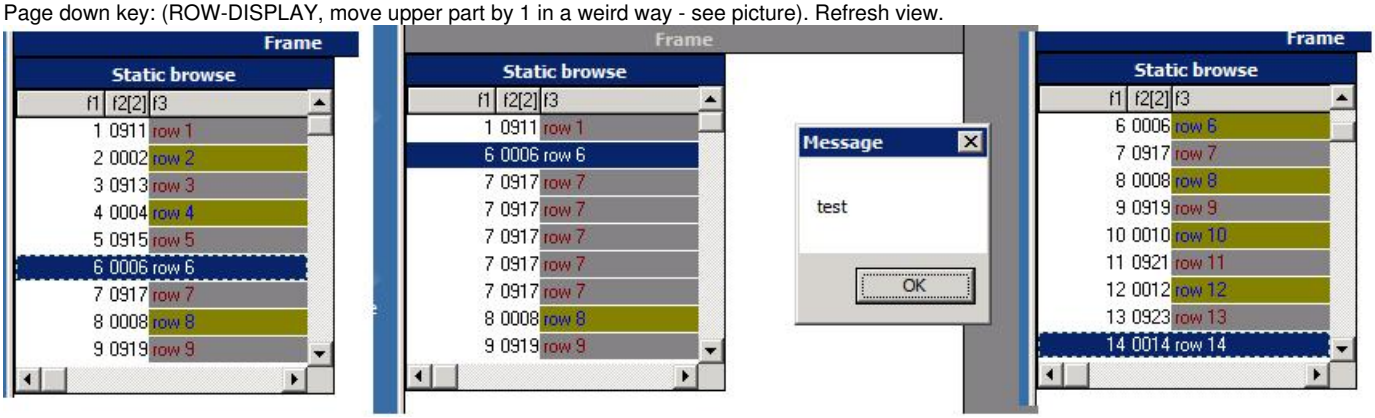
Yes, you should fix this. Do let me know if there is some negative implication or something else I'm missing.

#30 - 06/06/2017 10:42 AM - Stanislav Lomany

- File brws-pgdown.png added

Greg, I took a look how ROW-DISPLAY behaves when multiple new rows are displayed at once:

1. Mouse wheel scrolling: iteratively (ROW-DISPLAY, scroll by 1) scroll step times.
2. End, Home keys, thumb dragging, scroll to current row outside the view: trigger ROW-DISPLAY for the rows of the new view. Refresh view.
3. Search by key: If target row is before the end of the result set: (ROW-DISPLAY, scroll by 1) until target row is found. Trigger ROW-DISPLAY for all rows in view (again!). If target row is before the current row: 1) iteratively (ROW-DISPLAY, scroll by 1) scroll to the last row 2) trigger ROW-DISPLAY for the rows on the first page, refresh view 3) iteratively (ROW-DISPLAY, scroll by 1) scroll to the desired view.
4. Page down key: (ROW-DISPLAY, move upper part by 1 in a weird way - see picture). Refresh view.



5. Page up key: (ROW-DISPLAY in reverse order, move upper part by 1 in the same weird way). Refresh view.

I think it's not worth to exactly duplicate this behavior until (if) we come to the point when it will be necessary. I suggest to make minimal changes that will implement one kind of behavior: trigger ROW-DISPLAY for the new rows (non-iteratively), refresh view. Do you agree?

#31 - 06/08/2017 04:13 AM - Greg Shah

> I think it's not worth to exactly duplicate this behavior until (if) we come to the point when it will be necessary. I suggest to make minimal changes that will implement one kind of behavior: trigger ROW-DISPLAY for the new rows (non-iteratively), refresh view. Do you agree?

What is the effort to get this to match the weird Progress behavior?

As strange as the behavior is, because it is about event generation/trigger execution, it:

- can have a big effect on application behavior
- can be very hard to debug later when a customer app is broken

For these reasons I'm inclined to implement this now. It seems like you've aleady done the hard part of figuring out the behavior. Is there more investigation needed?

#32 - 06/08/2017 04:26 PM - Stanislav Lomany

> What is the effort to get this to match the weird Progress behavior?

For a start:

1. PageDown for some cases should be "incremental scrolling" rather than "scroll to row" - should I adjust that?
2. In order to search by key, we have to trigger ROW-DISPLAY for all rows before the found row. If search failed, we have to trigger ROW-DISPLAY for ALL rows in the result set. In our architecture straightforward solution assumes that we pass all rows to the client. Performance-wise that's terrible. What can we do here?

#33 - 06/09/2017 08:34 AM - Stanislav Lomany

All ROW-DISPLAY rules will be stored in updatable item 30.

We should decide if we want to support 1. iterative ROW-DISPLAY triggering (note that most of browse attributes, like DOWN or SCREEN-VALUE (not readable), are *not* accessible in 4GL ROW-DISPLAY trigger code). 2. ROW-DISPLAY triggers for non-displayed rows.
I understand that we want authentic triggering, but for user need which is:

1. set cell text
2. set cell decoration
3. get value of the current row

our approach seems to be is fine considering implementation time costs and performance problems. It's not a problem to add missing types of row triggering later. Thoughts?

#34 - 06/09/2017 09:12 AM - Stanislav Lomany

My bad, REFRESH doesn't trigger ROW-DISPLAY, only search by key. I'll clean up records.

#35 - 06/13/2017 09:33 AM - Greg Shah

> iterative ROW-DISPLAY triggering (note that most of browse attributes, like DOWN or SCREEN-VALUE (not readable), are not accessible in 4GL ROW-DISPLAY trigger code)

Actually, I wonder if this isn't the thing that helps us on the performance issue. It seems like the 4GL itself is in a "special mode" at this point. If common attributes are not allowed to be accessed, then it is definitely a special case. They are iteratively calling the trigger, but it seems like the normal conditions are (at least partially) suspended.

Can we iteratively execute the trigger on the server side, without down-calls to the client? We already know that the client has raised an event that puts us in this special iterative mode, right? And the client's state is not being fully accessed or synchronized (even in the 4GL). It seems we can do the processing on the server.

Thoughts?

#36 - 06/13/2017 10:08 AM - Stanislav Lomany

> Can we iteratively execute the trigger on the server side, without down-calls to the client?

At first look I think we can do that. But in the case trigger contains user interactions or alert boxes, we have to somehow detect that we have an interaction, send state to client and redraw browse. Is that a viable option?

#37 - 06/13/2017 10:40 AM - Greg Shah

> But in the case trigger contains user interactions or alert boxes, we have to somehow detect that we have an interaction, send state to client and redraw browse. Is that a viable option?

Yes, I think so. The down-call will naturally send any accumulated state. If browse-specific state in the screen buffer can be changed then we need to make sure that the screen buffer is sent. Also, forcing the redraw will need some logic. But the idea seems OK.

#38 - 06/13/2017 11:08 AM - Stanislav Lomany

> If browse-specific state in the screen buffer can be changed then we need to make sure that the screen buffer is sent.

Note that by "data" I mean all rows in the current view. I'll check what can be done more thoroughly.

#39 - 06/14/2017 05:17 PM - Stanislav Lomany

Some notes on what can be done during ROW-DISPLAY processing for multiple rows:

1. Browse itself and other widgets are NOT redrawn during ROW-DISPLAY processing. However browse scroller's thumb size/position is updated.
2. Input blocking statements are not allowed.
3. Messages in message area are displayed as usual.
4. The only way I found (so far) to pause processing, redraw widgets and display browse intermediate row state is an alert box (including error boxes).

Note that if we strive for 100% authenticity server-side ROW-DISPLAY doesn't make sense because we need to redraw thumb.

#40 - 06/14/2017 05:22 PM - Greg Shah

> Note that if we strive for 100% authenticity server-side ROW-DISPLAY doesn't make sense because we need to redraw thumb.

We can live with this one limitation for now. Make sure it is documented using a new Redmine issue that is linked here.

#41 - 06/15/2017 05:54 AM - Stanislav Lomany

Greg, am I right that server-side ROW-DISPLAY trigger should be called in that way?

1. Wrap BrowseWidget.getRows with the code from ThinClient.trigger.
2. Pass trigger-related info like trigger id and screen buffer as getRows parameters.
3. Call LogicalTerminal.trigger from getRows given number of times (if there are any rows to return).

#42 - 06/15/2017 08:26 AM - Greg Shah

> Wrap BrowseWidget.getRows with the code from ThinClient.trigger.

What do you mean by "wrap"? I prefer not to duplicate code, refactoring for common code is preferred. BrowseWidget is server side, but ThinClient.trigger() is on the client, so the idea is not clear.

Will the events that cause firing of the ROW-DISPLAY always be generated on the client?

#43 - 06/15/2017 08:33 AM - Stanislav Lomany

Basically my question is if the client-side trigger processing in ThinClient.trigger necessary for its counterpart in LogicalTerminal.trigger to work properly?

#44 - 06/15/2017 08:53 AM - Greg Shah

I'm not sure. Some of the processing in ThinClient.trigger() is to properly enable recursion, which you said cannot happen. Other processing is there to ensure the proper resumption of processing upon return. That processing might be needed.

#45 - 06/15/2017 09:08 AM - Stanislav Lomany

OK, I'll figure out which part is necessary. Overall I'm going to stick to the plan

ThinClient.trigger initial part (if necessary) -> getRows -> LogicalTerminal.trigger * row number times in getRows loop -> ThinClient.trigger finalizing part (if necessary).

Also:

1. If processing is paused in the middle of the trigger execution, we should provide to client side an intermediate set of rows to display in browse.
2. Server-side trigger processing might cause redrawing of other widgets, which doesn't happen in 4GL. Should I skip this differences and add to to the "thumb redrawing" redmine issue?

   I prefer not to duplicate code

Sure.

#46 - 06/15/2017 09:10 AM - Greg Shah

> Should I skip this differences and add to to the "thumb redrawing" redmine issue?

Yes, so long as the differences do not appear to be a problem for the user.

#47 - 06/21/2017 09:24 AM - Stanislav Lomany

It is interesting that there is no "iterative" ROW-DISPLAY triggering in ChUI. Except UP/DOWN keys which scroll by one row, other case follow the logic "trigger ROW-DISPLAY for the rows of the new view, refresh view".

#48 - 06/28/2017 05:32 PM - Stanislav Lomany

About "value cannot be displayed using ... " case for the browse:
If a cell value cannot be displayed using column format, the error message is displayed. In 4GL it is displayed once, while in P2J cells are rendered in draw(), which causes 1. excessive messages 2. abend because a message cannot be displayed from draw().
At first I thought that during the first rendering I could mark invalid values to avoid future rendering for this cell, but "value to string" functions in P2J return rendered value OR display error message and returns question marks, so there is no way to find out if an error has happened.
I think we can render a cell only once and cache the rendered string. On the bright side this will also speed up drawing.
The question is if we need the original values at all? I suppose that, except for the editing browse - no, we don't need it. It is also used in P2J through DataContainer.getValue, but VALUE attribute is not accessible for browses and columns in 4GL. But I still think to keep the values in order to make less changes and for the case I'm missing something, they don't take much memory.
What are your overall thoughts?

#49 - 06/29/2017 02:09 PM - Stanislav Lomany

- Related to Bug #3306: Improve intermediate state in ROW-DISPLAY trigger. added

#50 - 07/03/2017 05:19 AM - Greg Shah

> But I still think to keep the values in order to make less changes and for the case I'm missing something, they don't take much memory.

Agreed, keep the values in case they are needed.

#51 - 07/04/2017 06:25 PM - Stanislav Lomany

Interesting enough, if SCREEN-VALUE was updated in ROW-DISPLAY trigger during search by key, the new value is used for matching.

#52 - 07/08/2017 01:42 PM - Stanislav Lomany

Greg, I have a question about displaying browse intermediate state. The stack looks like

```
get rows
   trigger
      message box
         draw browse
            get rows for intermediate state
```

However I don't like the idea of getting rows in draw. The only thing that comes to my mind is, because AFAIK message box is the only way to display intermediate state of a browse, to add code to message box server-side function that notifies the browse in ROW-DISPLAY trigger state to push intermediate data to the browse. Any better ideas?

#53 - 07/08/2017 01:48 PM - Greg Shah

Don't we eventually need to store the state for synchronization to the client side? Can we store it as we go and then pick it up automatically the next time we call down to the client? I guess this might cause more updates than are done in the 4GL which we don't want to do.

My concern is purely that I prefer not to have browse-specific code in the message-box code. Other than that, it seems like a reasonable approach. To the degree that you can make it less browse-specific, it would be better.

If you go down this pat, make sure to add a comment to explain why that code is in message-box.

#54 - 07/08/2017 04:39 PM - Stanislav Lomany

> Don't we eventually need to store the state for synchronization to the client side? Can we store it as we go and then pick it up automatically the next time we call down to the client? I guess this might cause more updates than are done in the 4GL which we don't want to do.

Normally ROW-DISPLAY triggers modify cell attributes which in the current implementation requires no call downs to the client (at least it shouldn't). The problem is how to do detect that we are calling client to display something, and pass data at that point. Is there some magical place in the code that could help us?

> My concern is purely that I prefer not to have browse-specific code in the message-box code.

Mine too.

Also, I found another way to display intermediate state: PAUSE statement.

#55 - 07/10/2017 08:19 AM - Greg Shah

I guess the best way is to make the notification abstracted so that it is not browse specific, but in this case it will only be used by browse.

Then both message-box and pause can generate the notification.

#56 - 07/12/2017 05:15 AM - Stanislav Lomany

I took a look into customer's code and found that in one place non-string value is used as the initial value for ADD-CALC-COLUMN:

```
ADD-CALC-COLUMN("INTEGER":U, ">9":U , 0, "Page Seq.":U).
```

Documentation says that the initial value is a "character expression". Specifying non-character expression can cause error messages in 4GL, but it works fine for "0".

#57 - 07/14/2017 12:04 PM - Stanislav Lomany

> I suspect we need to dynamically convert the VALEXP/VALMSG from the schema at runtime ...
>
> The only real tricky part I see here is if these expressions can reference variables, widgets and other arbitrary state in the containing business logic. ... Please test this to see if it is needed.

Schema-level validation expressions for dynamic browse, unlike the static browse case cannot contain CAN-FIND, references to widgets, business logic functions, local buffers. Error message says that validation expression can contain only references to buffers known to browsed query, but I didn't succeed to create one, only the validated buffer could be referenced.

#58 - 07/25/2017 12:32 PM - Stanislav Lomany

Greg, consider the case of validation for DISPLAY .. ENABLE ALL case. During processing of ENABLE field we attach validation statements for each enabled field. However ENABLE ALL case is more sophisticated because

1. All validation statements are attached at progress.g. But when we meet a DISPLAY field token and can attach validation statement, we don't know when if it is ENABLE ALL case or not.
2. Processing of validation statements is performed in *validation.rules.

My solution is to:
1. attach validation statements in progress.g
2. very early in rules processing delete validation statements if browse doesn't have ENABLE ALL. That avoids creation of extra buffers that could be used in validation statements.
3. process as usual in *validation.rules.
Can you suggest a cleaner way?

#59 - 07/25/2017 12:36 PM - Greg Shah

Eric did the most recent validation processing implementation including the more sophisticated approach.

Are you simply suggesting that the only change would be very early in rules processing delete validation statements if browse doesn't have ENABLE ALL.?

Eric: Please review #3275-58 and comment.

#60 - 07/25/2017 12:43 PM - Stanislav Lomany

> Are you simply suggesting that the only change would be very early in rules processing delete validation statements if browse doesn't have ENABLE ALL.?

I just don't like the idea adding something that will be deleted later and wondering if I'm missing some lexer capabilities or if I went wrong conceptually.

#61 - 07/25/2017 01:23 PM - Greg Shah

- Related to Bug #2799: browse widget doesn't execute validation expressions in P2J (both chui and gui) added

#62 - 07/25/2017 01:32 PM - Greg Shah

> I just don't like the idea adding something that will be deleted later and wondering if I'm missing some lexer capabilities or if I went wrong conceptually.

This multi-part validation processing was designed this way on purpose. We don't want to deviate from it unnecessarily. For static widgets, the validation processing is quite tricky because it actually gets parsed at different scopes in the code depending on which language statements are in use (and their options). The ENABLE ALL case is the hardest because it means we have to track quite a bit of other state to know if validation expressions must be honored there (since there is no explicit list of widgets in this case). See #2692 for some details of Eric's work.

The lexer would not be involved here. The issue is that we need to dynamically add the validation expressions at parse time depending on the factors noted in #2692.

In regard to whether you are on the right track or not, we would have to see the code. Can you make a list of specific changes in 2892a to which you refer?

#63 - 07/25/2017 01:37 PM - Eric Faulhaber

Greg Shah wrote:

> Eric did the most recent validation processing implementation including the more sophisticated approach.
>
> Are you simply suggesting that the only change would be very early in rules processing delete validation statements if browse doesn't have ENABLE ALL.?
>
> Eric: Please review #3275-58 and comment.

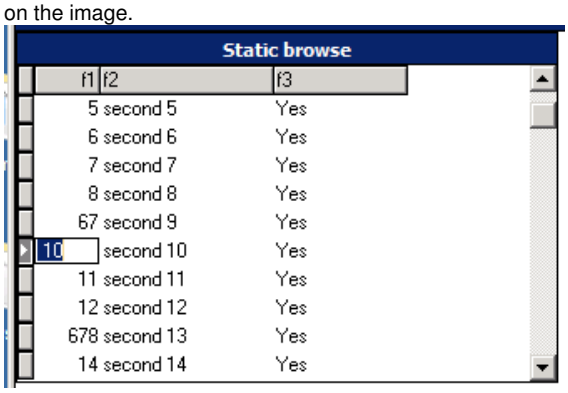
Based on what I remember of my earlier work in this area, I'm ok with the suggested approach. I think it would unnecessarily complicate the parser (which already got a lot more complicated in this area with the earlier changes) to try to differentiate the approach at this early stage. I think it makes sense to let the parser do its work attaching the validation statements, and then refine the results for browse purposes downstream in TRPL code, when more information is available.

#64 - 08/07/2017 08:31 AM - Stanislav Lomany

- File brws-auto-return.p added

- File auto-return-br-bug.png added

I found a bug related to AUTO-RETURN in original 4GL environment. Testcase attached, reproduction is DOWN to 14, UP to 9, type "678", result is on the image.



#65 - 08/07/2017 10:37 AM - Stanislav Lomany

Greg, I want to clarify if "WIDTH-PIXELS (conversion only)" I need to implement is an option that can be specified for a column and NOT the attribute with the same name.

#66 - 08/07/2017 10:40 AM - Greg Shah

> I want to clarify if "WIDTH-PIXELS (conversion only)" I need to implement is an option that can be specified for a column and NOT the attribute with the same name.

Yes, it is the option.

Is there anything needed for support of the attribute of that name? I think it may already be working.

#67 - 08/07/2017 11:03 AM - Stanislav Lomany

> Is there anything needed for support of the attribute of that name? I think it may already be working.

We need to re-layout browse on update. That may take some time.

#68 - 08/07/2017 11:55 AM - Greg Shah

> We need to re-layout browse on update. That may take some time.

We will need to do this, but getting all the conversion changes into the trunk is the priority. If that means the use of a second branch for the rest of the runtime changes, it is OK.

#69 - 08/07/2017 12:04 PM - Stanislav Lomany

Should I return to WIDTH-PIXELS attributes after I got all other options completed?
Should I go for VIEW-AS TOGGLE-BOX or validation stuff after I complete WIDTH-PIXELS option?

#70 - 08/07/2017 12:14 PM - Greg Shah

> Should I return to WIDTH-PIXELS attributes after I got all other options completed?

Do the conversion now. Runtime after validation is complete.

> Should I go for VIEW-AS TOGGLE-BOX or validation stuff after I complete WIDTH-PIXELS option?

Yes. We need the conversion features in the trunk ASAP. Runtime on all of these can be done next.

#71 - 08/10/2017 10:05 AM - Greg Shah

Please update the gap analysis rules (rules/gaps/*.rules) for the changes you have made here (attributes, methods, options).

#72 - 08/11/2017 01:34 PM - Greg Shah

Please also update gap marking for the AT and SCROLLBAR-VERTICAL browse options. It was recently proven that they are both fully supported at both conversion and runtime.

#73 - 08/11/2017 01:40 PM - Greg Shah

Please add the following to the work from this task.

SCROLL-TO-SELECTED-ROW method
NO-EMPTY-SPACE option (it is FIT-LAST-COLUMN without the ability to shrink last column)
ROW-HEIGHT-PIXELS option

#74 - 08/15/2017 02:44 PM - Greg Shah

Code Review Task Branch 2892a Revision 11189

Overall, this is a really good update.

Hynek: Please review the changes to BaseEntity, GenericWidget, BrowseWidget, BrowseColumnWidget.

1. In frame_scoping.rules, line 4734, the matching on !ancestor(prog.define_browse, -1) doesn't only exclude validation processing. This seems too broad. It will match any expressions that define columns in DISPLAY or otherwise specify options (format phrase, browse options). I think this needs to check specifically for the validate keyword.

2. I would have preferred for you to place the kw_set_s_ar and other methods in the load_descriptors in methods_attributes.rules.

3. Browse option kw_view_as has partial conversion support now. The gap marking doesn't show this.

4. The added client-side dependencies on ui/client/chui|gui/ in the server-side BrowseColumnWidget and BrowseWidget suggest these dependencies (like *uiCellAttributes) should reside in the ui package. There is nothing client-specific in these classes.

5. BrowseWidget.rowHeightError() should be at the bottom of the class with private methods.

6. In BrowseRow and CellAttributes, the implements clause should be on the next line.

7. In ChuiCellAttributes and GuiCellAttributes, the extends clause should be on the next line.

#75 - 08/15/2017 05:32 PM - Hynek Cihlar

Greg Shah wrote:

> Code Review Task Branch 2892a Revision 11189

Overall, this is a really good update.

Hynek: Please review the changes to BaseEntity, GenericWidget, BrowseWidget, BrowseColumnWidget.

I didn't find any issues in the changes.

#76 - 08/16/2017 04:47 AM - Stanislav Lomany

Issues from note 74 are fixed in branch 2892a revision 11190.

#77 - 08/16/2017 07:27 AM - Greg Shah

Code Review Task Branch 2892a Revision 11190

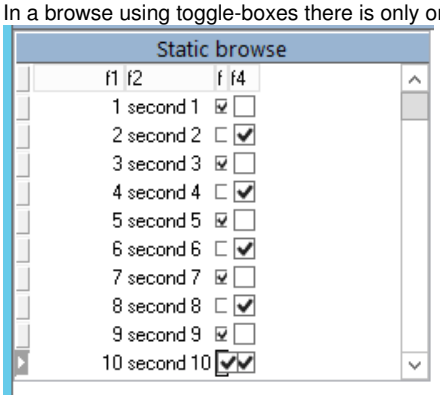
Everything looks good.

Go ahead with testing ChUI (conversion and runtime) and GUI (standalone tests, Hotel GUI, customer GUI).

#78 - 08/21/2017 10:18 AM - Stanislav Lomany

- File toggle-box.png added

In a browse using toggle-boxes there is only one real toggle-box control. Others are images.



#79 - 08/21/2017 05:49 PM - Stanislav Lomany

If a toggle-box is fit into a column with insufficient width, Progress 11.6 (win10 theme) scales toggle-box reducing its height. Progress 10.2 (classic theme) doesn't reduce height, but instead trims right and left parts. I'll go with the first option.

#80 - 08/22/2017 07:18 AM - Greg Shah

Stanislav Lomany wrote:

> If a toggle-box is fit into a column with insufficient width, Progress 11.6 (win10 theme) scales toggle-box reducing its height. Progress 10.2 (classic theme) doesn't reduce height, but instead trims right and left parts. I'll go with the first option.

This is something that should be delegated to the theme implementation. Each theme should match the behavior found. I doubt it is related to the OE version.

#81 - 08/22/2017 07:19 AM - Stanislav Lomany

2892a revision 11194 contains:

CURRENT-COLUMN
INDEX
COLUMN-READ-ONLY
ROW-HEIGHT-PIXELS
BUFFER-FIELD

Methods

ADD-CALC-COLUMN and ROW-DISPLAY event processing
SET-SORT-ARROW
CLEAR-SORT-ARROWS
SCROLL-TO-CURRENT-ROW
SCROLL-TO-SELECTED-ROW

Options

LABEL-FGCOLOR
VIEW-AS TOGGLE-BOX (conversion)
LABEL-FONT (runtime only)
COLUMN-FGCOLOR
NO-SCROLLBAR-VERTICAL (runtime only)
AUTO-RETURN (runtime only)
WIDTH-PIXELS (conversion only)

Revision 11196 (not yet regression tested) contains:

NO-EMPTY-SPACE
ROW-HEIGHT-PIXELS (option)

Will be implemented on this week:
VIEW-AS TOGGLE-BOX (runtime)
NO-VALIDATE
NO-AUTO-VALIDATE
NO-VALIDATE (runtime only)

Uncertain about estimates:
validation for dynamic browse

#82 - 08/22/2017 07:24 AM - Stanislav Lomany

How can I check out how win8 theme should look?

#83 - 08/23/2017 10:13 AM - Greg Shah

Code Review Task Branch 2892a Revision 11191

I'm fine with the changes.

#84 - 08/23/2017 10:16 AM - Greg Shah

Did both conversion and runtime ChUI regression testing pass for rev 11194/11196?

I don't think that ChUI regression testing is needed for the changes between 11194 and 11196. But for 11196, manual GUI testing is important. If that passes (or did pass already), then you can merge to trunk. Can you get this done ASAP?

#85 - 08/23/2017 10:47 AM - Greg Shah

The gap analysis rules need to be updated for the latest changes. ChUI conversion regression test will be sufficient to confirm they are OK.

#86 - 08/23/2017 03:44 PM - Stanislav Lomany

I'll update gap analysis and merge now.

#87 - 08/23/2017 05:30 PM - Stanislav Lomany

Created task branch 3275b from trunk revision 11159.

#88 - 08/25/2017 01:42 PM - Greg Shah

The browse option validate needs to be added to gap analysis marking.

#89 - 08/28/2017 10:35 AM - Stanislav Lomany

Toggle-box values of calc columns are NOT displayed in Progress 10.2, definitely it is a progress bug. Works as expected in Progress 11.6.

#90 - 08/28/2017 10:43 AM - Greg Shah

Stanislav Lomany wrote:

> Toggle-box values of calc columns are NOT displayed in Progress 10.2, definitely it is a progress bug. Works as expected in Progress 11.6.

You don't have to reproduce the bug.

#91 - 08/29/2017 10:28 AM - Greg Shah

In #3330 we found there is an undocumented usage of NUM-ENTRIES as a browse attribute. 3330a has conversion support for it (including stubs in BrowseInterface). Please include runtime support for NUM-ENTRIES in this task. It can be done when the rest of the validation work is finished.

#92 - 09/01/2017 03:21 AM - Stanislav Lomany

Rebased task branch 3275b from P2J trunk revision 11163.

#93 - 09/01/2017 08:31 AM - Stanislav Lomany

When you have time, please review task branch 3275b rev 11169.

#94 - 09/06/2017 12:07 PM - Greg Shah

Code Review Task Branch 3275b Revision 11169

The changes look good.

1. Browse.getCellTextNoError() should be in the UI package. We don't want ui/client/ dependencies in the ui/ package (BrowseRow, BrowseWidget). The problem in this case is that our DisplayFormat classes are deep inside the client. They probably should be made to be generic (and not client-specific). At this point, you can leave it, but I just want to highlight that it is not the way we would prefer it.

2. Browse.createEditor() can only create a FILL-IN, even though it is described as creating the properly configured widget type.

3. Calls to Browse.getActiveFillIn() and Browse.getActiveToggleBox() assume that the caller only calls these if they have checked Browse.isEditorFillIn() or Browse.isEditorToggleBox() first. Otherwise there will be a ClassCastException. The Javadoc for the getActive*() methods must include a much more explicit warning of what must be done and the consequence of not doing it (throws ClassCastException).

4. BrowseImpl.draw(), BrowseImpl.drawCaret() call getActiveFillIn() with no protection call to isEditorFillIn(). If only FILL-IN is allowed in ChUI, then it should be made clear that is the case.

5. What is your estimate of the effort needed to implement COMBO-BOX?

#95 - 09/11/2017 01:35 PM - Stanislav Lomany

> 5. What is your estimate of the effort needed to implement COMBO-BOX?

I think conversion + runtime will take about a week.

#96 - 09/11/2017 02:34 PM - Greg Shah

- Related to Feature #2628: Non-fill-in column support in browse. added

#97 - 09/11/2017 02:34 PM - Greg Shah

For now we will leave combo-box support for #2628.

#98 - 09/13/2017 06:22 AM - Stanislav Lomany

There are multiple browse issues that has been found during testing:

1. Scroll by mouse wheel doesn't work if the pointer is positioned over an empty area to the right to the columns or over an editor.

2. Column and row resizing doesn't work properly:
   2.1 when the pointer is positioned below the row separation line, the lower row is resized instead of the upper one;
   2.2 editing is stopped on row resize;
   2.3 if rows are thin, resize area (where pointer changes) is smaller than required;
   2.4 after row height is increased, browse cannot be scrolled with keys or scrollbar + abend on scrolling with mouse wheel;
   2.5 if editing is active, columns are not resizable.
3. Down, up, shift-tab keys do not work for toggle-box editors.
4. In ChUI:
   4.1 for an editable browse double Ctrl-G causes abend;
   4.2 browse do not start editing when we tab back into it.

#99 - 09/13/2017 04:03 PM - Stanislav Lomany

I just realized that validation statements for browse are not properly converted when it comes to the validated variable. E.g. instead of

```
((ValidationExpr<integer>) (integer bookId) -> isNotEqual(book.getBookId(), 33)), "Cannot be 33!"
```

it should be

```
((ValidationExpr<integer>) (integer bookId) -> isNotEqual(bookId, 33)), "Cannot be 33!"
```

Working on it.

#100 - 09/15/2017 04:20 AM - Stanislav Lomany

Regression testing for 3275b passed. The only thing is that IIRC tc_job_002 was failing with

```
tc_job_002: failure in step 40: 'Unexpected EOF in actual at page # 1.'
```

and now it is

```
tc_job_002: failure in step 41: 'Line size mismatch (line # = 11, base = 0, actual = 91).'
```

Was baseline or harness updated?
Should I check 3275b into trunk?

#101 - 09/15/2017 09:05 AM - Greg Shah

Constantin/Eugenie/Ovidiu: Please see #3275-100. Does anyone have a reason for why the tc_job_002 now gets past step 40?

#102 - 09/15/2017 09:07 AM - Greg Shah

> Should I check 3275b into trunk?

We are trying to get 3330a and 3222a into the trunk as priorities. However, the 3330a code still has two regressions to fix. Considering your code has passed testing, I think it is safe to merge 3275b into 3330a.

#103 - 09/15/2017 10:40 AM - Stanislav Lomany

3275b was merged into 3330a as revision 11238.

#104 - 09/15/2017 11:25 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> Constantin/Eugenie/Ovidiu: Please see #3275-100. Does anyone have a reason for why the tc_job_002 now gets past step 40?

My last tests stopped at 41 with Line size mismatch (line # = 11, base = 0, actual = 91), too.

#105 - 09/15/2017 12:03 PM - Constantin Asofiei

Greg Shah wrote:

> Constantin/Eugenie/Ovidiu: Please see #3275-100. Does anyone have a reason for why the tc_job_002 now gets past step 40?

The problem is with unix2dos: see #3264-16 and subsequent

#106 - 09/15/2017 01:41 PM - Greg Shah

Somehow I didn't notice that the failing step was now 41 not 40. I guess that came with your MAJIC update.

#107 - 09/19/2017 03:36 PM - Greg Shah

- Related to Bug #3335: buggy no-validate in browse widget added

#108 - 09/20/2017 06:12 AM - Stanislav Lomany

Created task branch 3275c from P2J trunk revision 11164.

#109 - 09/29/2017 11:52 AM - Stanislav Lomany

Guys, when I try to convert a procedure using the code from DynamicQueryHelper.parse, expressions lose their first element. E.g.

```
message "test".
```

is converted to

```
package com.goldencode.p2j.persist.dynquery;

import com.goldencode.p2j.util.*;

import static com.goldencode.p2j.util.BlockManager.*;

public class DynGenQuery
{
   public void execute()
   {
      externalProcedure(DynGenQuery.this, new Block((Body) () ->
      {
         "test";
      }));
   }
}
```

This happens on CORE_CONVERSION phase. Do you have any ideas off the top of your heads?

#110 - 09/29/2017 12:12 PM - Greg Shah

I can't reproduce your results with trunk.

This program (testcases/uast/simple_message.p):

```
message "Hello World!".
```

When converted the code is this:

```
package com.goldencode.testcases;

import com.goldencode.p2j.util.*;
import com.goldencode.p2j.ui.*;

import static com.goldencode.p2j.util.BlockManager.*;
import static com.goldencode.p2j.ui.LogicalTerminal.*;

/**
 * Business logic (converted to Java from the 4GL source code
 * in simple_message.p).
 */
public class SimpleMessage
{
   /**
    * External procedure (converted to Java from the 4GL source code
    * in simple_message.p).
    */
   public void execute()
   {
      externalProcedure(SimpleMessage.this, new Block((Body) () ->
      {
         message("Hello World!");
      }));
   }
}
```

There must be something else in your testscase OR there is some regression in your branch.

#111 - 09/29/2017 12:15 PM - Constantin Asofiei

Stanislav Lomany wrote:

> Guys, when I try to convert a procedure using the code from DynamicQueryHelper.parse, expressions lose their first element. E.g.

The key here is DynamicQueryHelper.parse was supposed to parse only 4GL query code, not generic 4GL code. If you look into conver/core_conversion.xml, rule-sets have load-condition="runtime-query" attributes - DynamicQueryHelper.parse will use ONLY the rule-sets marked as runtime-query.

But, OTOH, why do you need this? Do we really want to be able to parse generic 4GL code at runtime? Because (at least in query case), if we include other rule-sets, there will be a performance penalty.

#112 - 09/29/2017 12:23 PM - Greg Shah

Sorry, I missed the most important part of your message (the DynamicQueryHelper.parse() part).

Constantin's point about the missing conversion rule sets is correct. BUT I'm also sure that we don't need to parse language statements, blocks and other general 4GL code. You really only need to parse the expr rule in progress.g.

Before you "go there" with using DynamicQueryHelper, I do wonder the following: does the 4GL itself implement enough restrictions on what is valid for runtime processing in a dynamic browse, that we can convert the expressions **in advance** at conversion time and just provide the necessary variable references as parameters at runtime?

#113 - 09/29/2017 01:51 PM - Stanislav Lomany

> convert the expressions **in advance** at conversion time and just provide the necessary variable references as parameters at runtime?

There is buffer-field:VALIDATE-EXPRESSION attribute which allows you to set an expression at runtime, so there is no point of making separate facility for schema-level validation expressions.
BTW, do we need to implement VALIDATE-EXPRESSION?

#114 - 09/29/2017 01:58 PM - Eric Faulhaber

Stanislav Lomany wrote:

> BTW, do we need to implement VALIDATE-EXPRESSION?

In the current project, there are two uses of this attribute, in one file. They both set the expression to empty string.

#115 - 09/29/2017 02:32 PM - Greg Shah

Eric Faulhaber wrote:

> Stanislav Lomany wrote:
>
> > BTW, do we need to implement VALIDATE-EXPRESSION?

> In the current project, there are two uses of this attribute, in one file. They both set the expression to empty string.

Presumably this disables validation for the field. What about the schema-validation expressions? Do dynamic browses honor them? Converting them in advance would be a workable solution.

In other words: what is needed here?

For future reference, I'd like to know you really implement any expression and assign it to VALIDATE-EXPRESSION. I would doubt it.

#116 - 09/29/2017 04:18 PM - Stanislav Lomany

> What about the schema-validation expressions? Do dynamic browses honor them?

Yes, but the expressions have limitations described in #3275-57

#117 - 09/30/2017 04:59 PM - Stanislav Lomany

> Converting them in advance would be a workable solution.

So, are you suggesting to add some validator classes with converted functions like

```
function do-validate returns logical (buffer book for book, input book-id as integer):
   return string(book-id) <> book-title. // validation statement here
end.
```

under dmo package?

> For future reference, I'd like to know you really implement any expression and assign it to VALIDATE-EXPRESSION. I would doubt it.

Sorry, what do you mean?

#118 - 10/01/2017 09:04 AM - Greg Shah

Stanislav Lomany wrote:

> What about the schema-validation expressions? Do dynamic browses honor them?

Yes, but the expressions have limitations described in #3275-57

#3275-57 mentions some things that cannot be included. But it is not clear about the things that are included. For example, can variables from the containing business logic be referenced?

#119 - 10/01/2017 09:13 AM - Greg Shah

> Converting them in advance would be a workable solution.

So, are you suggesting to add some validator classes with converted functions like

Yes, something like this. Since we know the schema validation expressions in advance, it seems like a static conversion could work. I don't know that they can be placed in the dmo directory because they can have references to business logic state.

Can you statically determine (at conversion time) which tables are referenced from a dynamic browse?

> For future reference, I'd like to know you really implement any expression and assign it to VALIDATE-EXPRESSION. I would doubt it.

Sorry, what do you mean?

I meant to say "I'd like to know if you can really implement any expression and assign it to VALIDATE-EXPRESSION.". Do the same limits from #3275-57 apply to those expressions?

#120 - 10/01/2017 05:38 PM - Stanislav Lomany

> Can you statically determine (at conversion time) which tables are referenced from a dynamic browse?

No, an arbitrary query handle is assigned to a dynamic browse.

> I meant to say "I'd like to know if you can really implement any expression and assign it to VALIDATE-EXPRESSION.". Do the same limits from #3275-57 apply to those expressions?

The same limits apply.

> But it is not clear about the things that are included.

According to the error message it can be constants and buffer/field references to buffers known to the query. According to testing it can be constants and buffer/field references to the validated buffer only.

#121 - 10/02/2017 07:42 AM - Greg Shah

> According to the error message it can be constants and buffer/field references to buffers known to the query. According to testing it can be constants and buffer/field references to the validated buffer only.

There is no way to reference anything (including variables) in the business logic?

#122 - 10/02/2017 07:44 AM - Greg Shah

In regard to browse-handle:VALIDATE-EXPRESSION = "". did you check what this does?

#123 - 10/02/2017 10:25 AM - Stanislav Lomany

> There is no way to reference anything (including variables) in the business logic?

Right.

> In regard to browse-handle:VALIDATE-EXPRESSION = "". did you check what this does?

Disables validation.
Note that validation expressions are applied only if they set *before* adding corresponding column to the browse.

#124 - 10/02/2017 10:37 AM - Greg Shah

Then it seems to me that the DynamicQueryHelper is already quite close to supporting this feature. We will need to convert at runtime but since there is no access to the state of the business logic, it is a pretty good match already.

Instead of generating the query tree, you would generate a validator. The interpreter probably already supports all the features you need.

Do you have any open questions on this?

#125 - 10/02/2017 03:24 PM - Stanislav Lomany

> Do you have any open questions on this?

Well, I thought to wrap a validation statement in a function or procedure like this:

```
function do-validate returns logical (buffer book for book, input book-id as integer):
   return string(book-id) <> book-title. // validation statement here
end.
```

and then run it to validate. But it cannot be converted using the current current set of runtime conversion rules. Should I work on AST in order to make it convertible by the current runtime set in core_conversion.xml, or maybe can have two different conversion profiles for queries and validation statements?

#126 - 10/02/2017 03:28 PM - Constantin Asofiei

Stanislav Lomany wrote:

> Should I work on AST in order to make it convertible by the current runtime set in core_conversion.xml, or maybe can have two different conversion profiles for queries and validation statements?

Is better to add a new marker, i.e. runtime-valexp; you can separate them by comma, so you will have runtime-query,runtime-valexp if a rule-set is enabled for both. This way will not interfere with the query-related stuff.

#127 - 10/02/2017 03:46 PM - Greg Shah

> Well, I thought to wrap a validation statement in a function or procedure like this:

I think this is unnecessary. We don't need the 4GL function skeleton here. I think we only need the ability to convert and evaluate a logical expression. If the VAL-MSG can be dynamic, then we would need the ability to convert and evaluate a character expression.

Try to keep this as simple as possible. The Java runtime code can handle the proper setup and buffer handling and then call the the evaluate() method.

The interpreter already knows how to process a WHERE clause, which is just a logical expression after all. And it can have character subexpressions so much of that is already present too.

I don't know if any of that processing is different from WHERE clauses than it would be for 4GL expressions but I would expect it to be the same. The 4GL executes WHERE clauses in the same process with the business logic and the runtime code is the same if I understand correctly.

> Is better to add a new marker, i.e. runtime-valexp; you can separate them by comma, so you will have runtime-query,runtime-valexp if a rule-set is enabled for both. This way will not interfere with the query-related stuff.

Agreed, though I am hoping the validation expression can be simpler than the query approach.

#128 - 10/03/2017 02:16 PM - Stanislav Lomany

Interesting enough, in validation statements for a dynamic browse you cannot access fields of the validated buffer, except the validated field. Value of other fields id ?.

#129 - 10/03/2017 02:16 PM - Eric Faulhaber

Greg Shah wrote:

> The interpreter already knows how to process a WHERE clause, which is just a logical expression after all. And it can have character subexpressions so much of that is already present too.
>
> I don't know if any of that processing is different from WHERE clauses than it would be for 4GL expressions but I would expect it to be the same. The 4GL executes WHERE clauses in the same process with the business logic and the runtime code is the same if I understand correctly.

If you're referring to the RuntimeJastInterpreter, it's not dealing with WHERE clauses directly. The 4GL record retrieval construct containing a WHERE clause is converted into an instance of a specific class which implements the P2JQuery interface. The RuntimeJastInterpreter is interpreting that JAST directly and executing that query.

Ovidiu, perhaps you can provide some better guidance than I can in this area.

#130 - 10/03/2017 03:11 PM - Stanislav Lomany

I can convert something like

```
define buffer book for book.
return book-id > 5.
```

to

```
public class DynGenQuery
{
   Book.Buf book = RecordBuffer.define(Book.Buf.class, "p2j_test", "book", "book");

   public void execute()
   {
      externalProcedure(DynGenQuery.this, new Block((Body) () ->
      {
         RecordBuffer.openScope(book);
         returnNormal(isGreaterThan(book.getBookId(), 5));
      }));
   }
}
```

So if 1. replace reference to validated book.book-id with a variable. 2. extract expression under returnNormal we will get what we want:

```
public class DynGenQuery
{
   public logical validate(character bookId)
   {
```

```
      return isGreaterThan(bookId, 5));
   }
}
```

#131 - 10/03/2017 03:55 PM - Stanislav Lomany

Question is if we want to pass buffers to the validated function:
1. validated value definitely can be referenced and it is passed as a variable;
2. other fields of the validated buffer can be referenced, but they have unknown values, so they are useless and if a user uses validation statement containing them, IMO he should we warned;
3. other buffers known to the query cannot be referenced on practice (but one of the error messages suggests that they can).

#132 - 10/03/2017 04:00 PM - Greg Shah

For now, there is no good reason to process with the buffer. Use the variable approach.

> 2. other fields of the validated buffer can be referenced, but they have unknown values, so they are useless and if a user uses validation statement containing them, IMO he should we warned;

Please detect this case and then do the following:

- Output a warning to the console.
- Rewrite these other fields as the unknown value literal, which is the logical equivalent.

#133 - 10/04/2017 07:30 AM - Stanislav Lomany

> 3. other buffers known to the query cannot be referenced on practice (but one of the error messages suggests that they can).

OK, buffers can be referenced using functions like avail(tt2). Still, tt2.f3 of tt2:buffer-field('f3') cannot be used. I think that means we have to pass buffers to a validation function.

Also, I see 2-level aging cache for dynamic queries. What kind of caching should be used for validation statements? Note, they can be for temp tables too.

#134 - 10/04/2017 04:12 PM - Greg Shah

> I think that means we have to pass buffers to a validation function.

Yes.

> Also, I see 2-level aging cache for dynamic queries. What kind of caching should be used for validation statements? Note, they can be for temp tables too.

Probably the same approach makes sense.

Eric?

#135 - 10/04/2017 05:17 PM - Eric Faulhaber

Greg Shah wrote:

> > Also, I see 2-level aging cache for dynamic queries. What kind of caching should be used for validation statements? Note, they can be for temp tables too.
>
> Probably the same approach makes sense.
>
> Eric?

I don't know that the two-level cache design makes sense for validation expressions, but that may be because I'm not clear on what the validation expression JASTs will look like compared to the dynamic query JASTs.

Ovidiu, I'm going back over #2488 and the caching code in DynamicQueryHelper, and the implementation is a bit confusing to me now. Can you please comment on the primary difference between the two caching levels? As I recall, the distinction is about the caching of query parameters with the JAST, which may not be applicable for validation expressions.

#136 - 10/09/2017 07:55 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Ovidiu, I'm going back over #2488 and the caching code in DynamicQueryHelper, and the implementation is a bit confusing to me now. Can you please comment on the primary difference between the two caching levels? As I recall, the distinction is about the caching of query parameters with the JAST, which may not be applicable for validation expressions.

The the 2nd level was added in order to gain a bit of performance when converting similar (parametrizable) queries. For example, if FIND FIRST book WHERE book-id 555 is processed it will be stored in both caches: as it is for cache 1, and with constant 555 replaced with %p1 in cache lvl2. A subsequent call for same query will be a hit in cache level 1, but a call for FIND FIRST book WHERE book-id 444 will miss the first cache. We then do a quick syntactical analysis (parse only), and find the parametrised jast in 2nd level with key FIND FIRST book WHERE book-id == %p1. If the 2nd level is a miss, the full ANNOTATIONS / BASE_STRUCTURE / CORE_CONVERSION cycle is needed to process to obtain the jast.

For the moment, the DynamicQueryHelper and RuntimeJastInterpreter can handle only two very specific types of statements: simple FIND ... and OPEN QUERY ... queries. The validation expressions are some kind of functions - at least this is how I think the dynamic conversion should output them. A dedicated dynamic helper should be implemented using the structure of DynamicQueryHelper. However, RuntimeJastInterpreter will probably be able to process/interpret the resulting functions with minimum changes (if any), as it was design to handle complex parametrised jasts.

#137 - 10/09/2017 03:41 PM - Stanislav Lomany

Some questions for you:
1. Do we need to implement VALIDATION-EXPRESSION attribute? Even is it is only support for VALIDATION-EXPRESSION = "", that will require some additional time, because this attribute is field-specific rather than browse-specific.
2. So what about caching?
2.1 Two-level caching makes sense because there are similar validation expressions in schema. Like

```
index(""SC"",i-code) > 0
index(""IBGCWVF"",mat-type) > 0
```

However I do not extract parameters an this point, so adding this feature adds time to implementation. Should I go for it?
2.2 What is the key for first-level caching? It is

- validation statement,
- validated buffer (DMO class),
- validate field,
- other buffers handled by the same query - but their fields cannot be accessed (while the buffers can) - should I include them into the key for the case?

2.3 If we want to add full support for VALIDATION-EXPRESSION statement, then temp-tables can be validated. Should cache be scoped?

3. At this point the main code of validation statement parsing code contains around a half of code from DynamicQueryHelper.parse. Some other functions from DynamicQueryHelper are used as well. Should stuff related to validation statements reside in other class, like DynamicValidationHelper, and use functions of DynamicQueryHelper?

Side note: I'm using runtime/postprocess_open_query postprocessing, which may be an overkill, but it works and I want to leave it as is for now.

#138 - 10/09/2017 03:57 PM - Greg Shah

> 1. Do we need to implement VALIDATION-EXPRESSION attribute? Even is it is only support for VALIDATION-EXPRESSION = "", that will require some additional time, because this attribute is field-specific rather than browse-specific.

Yes, it is in use so it needs to be implemented.

The implementation can be limited to the "" usage.

> 2. So what about caching?

Eric may have another opinion. But to me, a 2nd level cache doesn't seem to be needed here.

> Should stuff related to validation statements reside in other class, like DynamicValidationHelper, and use functions of DynamicQueryHelper?

Yes, use common code but the validation-specific stuff should be separate.

#139 - 10/10/2017 07:13 AM - Stanislav Lomany

Should cache for validation statements be aged?

#140 - 10/10/2017 08:11 AM - Greg Shah

Yes. If possible, use the same technique already used for the queries.

#141 - 10/11/2017 06:51 AM - Stanislav Lomany

In BufferFieldImpl following functions:

```
changeColumnLabel
changeLiteralQuestion
changeValidateExpression
changeValidateMessage
```

work incorrectly because they change corresponding attributes globally rather than for the specific buffer. I suggest to create in RecordBuffer maps like Map<String, String>validateExpressions which keep updated attribute values for buffer fields. If there is no map entry, default global value is used. OK?

#142 - 10/11/2017 10:03 AM - Eric Faulhaber

Stanislav Lomany wrote:

> I suggest to create in RecordBuffer maps like Map<String, String>validateExpressions which keep updated attribute values for buffer fields. If there is no map entry, default global value is used. OK?

Are you saying "globally" because the updated values are changed in TableMapper.LegacyFieldInfo, rather than in a particular RecordBuffer instance? Yes, I guess your solution is ok. What would the map key strings represent, the DMO property name, or the legacy field name?

It should be fairly uncommon to change any of these attributes, so please only initialize the maps if a value is changed, otherwise leave them as null (rather than empty maps).

#143 - 10/11/2017 10:23 AM - Stanislav Lomany

> What would the map key strings represent, the DMO property name, or the legacy field name?

DMO property name.

#144 - 10/17/2017 05:06 PM - Stanislav Lomany

Rules that take place when a buffer-field attribute is assigned:
1. Attributes for permanent and temp-table buffers are buffer-specific, except the next case:
2. Attributes for default temp-table buffers are used for all other buffers for this temp-table unless these buffers have their attributes assigned. This kind of attributes take place while default buffer/table scope is active.

#145 - 10/20/2017 07:38 AM - Stanislav Lomany

According to the rules above, all attributes are normally buffer-specific. However there is the 4GL quirk which takes place when an unknown value is assigned to the LABEL attribute.
1. For *other* attributes I've dealt with, including COLUMN-LABEL, you cannot assign an unknown value to it, you get an error message.
2. When you assign an unknown value to LABEL, you are not getting an error message, and the actual assigned value is an empty string.
3. The quirk takes place only if no explicit LABEL was assigned to the target buffer. E.g.
quirk:

```
buffer book:buffer-field("book-id"):label = ?.
```

no quirk:

```
buffer book:buffer-field("book-id"):label = "Some label".
buffer book:buffer-field("book-id"):label = ?.
```

4. For temp tables, when LABEL is assigned to a buffer, it is also set for the default temp-table buffer as well (note the value of the default buffer's LABEL is used as the default value for other buffers). E.g.

```
def temp-table tt1 field f1 as integer label "Initial".
def buffer tt2 for tt1.
def buffer tt3 for tt1.
def buffer tt4 for tt1.

message "before: " + string(buffer tt1:buffer-field("f1"):label).

buffer tt1:buffer-field("f1"):label = "TT1".
buffer tt3:buffer-field("f1"):label = "TT3".
buffer tt2:buffer-field("f1"):label = ?.

message "tt1: " + string(buffer tt1:buffer-field("f1"):label) +
        "      tt2: " + string(buffer tt2:buffer-field("f1"):label) +
        "      tt3: " + string(buffer tt3:buffer-field("f1"):label) +
        "      tt4: " + string(buffer tt4:buffer-field("f1"):label).

/* output is:
before: Initial.
tt1:     tt2:     tt3: TT3        tt4:
*/
```

5. For permanent tables, assigning an unknown value to a buffer's LABEL sets the value for ALL buffers, except the ones which have it explicitly set. The effect lasts until the user context is active. So, basically, the effect is the same as for temp tables.

I think it is a 4GL bug, they forgot about unknown value protection or something like that. The effect is too broad, and not described in the documentation. This case can be easily detected, so I suggest to leave is as UnimplementedFeature. Do you agree?

#146 - 10/20/2017 07:43 AM - Greg Shah

This one seems straightforward to implement. Why not just implement it now while you are reworking the buffer-specific behavior?

Every unimplemented feature we leave behind is work to do later. If it is small work to get it done now, we should clear it. If it is more work, then we can defer it.

#147 - 10/20/2017 09:51 AM - Stanislav Lomany

No problems to implement, except that I need to make TableMapper.legacyFieldInfo() context-local for permanent tables. Do you see any potential problems with that?

#148 - 10/20/2017 10:03 AM - Eric Faulhaber

Stanislav Lomany wrote:

> No problems to implement, except that I need to make TableMapper.legacyFieldInfo() context-local for permanent tables. Do you see any potential problems with that?

Just the obvious issues of duplicating most of this information in memory and not taking advantage of caching this information across sessions in terms of performance. But if this is what we need to manage this info correctly, I don't see that we have much of a choice. At some point, we may need to refactor TableMapper, so that we are not doing a separate lookup to gather each bit of information. This performance weakness will be amplified when we substitute slower context-local lookups for the simpler map lookups we do today.

#149 - 10/20/2017 11:25 AM - Stanislav Lomany

I think it may be better to add context-local map where LABEL attributes modified by this quirk reside. That mean very small changes to TableMapper.

#150 - 10/20/2017 11:45 AM - Eric Faulhaber

This seems like a good compromise. Please go ahead with the change to TableMapper.

#151 - 10/24/2017 11:59 AM - Stanislav Lomany

1. In ChUI:
   4.1 for an editable browse double Ctrl-G causes abend;

Fixed.

> 4.2 browse do not start editing when we tab back into it.

Created #3361

#152 - 10/25/2017 07:53 PM - Stanislav Lomany

Rebased task branch 3275c from P2J trunk revision 11182.

#153 - 10/26/2017 07:33 AM - Greg Shah

Is 3275c ready for review?

#154 - 10/26/2017 07:33 AM - Greg Shah

- Related to Bug #3361: In ChUI browse does not start editing when we tab back into it added

#155 - 10/26/2017 07:53 AM - Stanislav Lomany

Yes, just finished. Please review 3275c revision 11193.

#156 - 10/26/2017 09:25 AM - Greg Shah

Code Review Task Branch 3275c Revision 11193

Eric: Please review the changes in the persist package.

Everything seems quite good to me. A couple of minor things:

1. In ControlEntity, the use of LogicalTerminal.message() instead of something in ErrorManager seems incorrect. In ErrorManager we deal with things like headless mode, no-error and other session-level state. Although I think that headless mode cannot be in use in this case, I worry that the other session state may be important.

2. Browse.leaveRow() needs javadoc.

#157 - 10/26/2017 01:42 PM - Eric Faulhaber

Greg Shah wrote:

> Eric: Please review the changes in the persist package.

Wow, that's a big set of changes! The update generally looks good to me from a persistence perspective. This will need ETF testing, as normal regression testing will not exercise the dynamic code conversion and interpretation code paths, and some of the changes (particularly to DynamicQueryHelper) are significant.

Ovidiu, please review the changes specific to the dynamic compilation and runtime JAST interpretation code. The goal was to add dynamic conversion and interpretation for validation expressions in the same way we do for queries.

Some comments:

- Please replace the hard tab at line 645 in build.xml.
- In DynamicQueryHelper.injectVariable, why are all literal values replaced with the same variable?
- In RecordBuffer.setFieldLabel, please change the comment "trigger quirk" to something more explanatory. Currently, you have to look at the corresponding TableMapper changes to fully understand what this means, and the connection between these related changes is not obvious. It is ok to duplicate the explanation from TableMapper; I found the comments in that class more helpful to understand the situation.
- I realize I am guilty of this myself in places, and we don't have time to fix this now, but for future development I want to discourage this example of an anti-pattern: instead of using isTemporary within the new RecordBuffer methods to conditionally branch logic, we should implement methods in the parent class (in this case RecordBuffer) with the default behavior and override them in the subclass (in this case TemporaryBuffer) to change that behavior.

#158 - 10/26/2017 02:05 PM - Stanislav Lomany

- In DynamicQueryHelper.injectVariable, why are all literal values replaced with the same variable?

I needed a function to replace multiple given nodes with the same variable. Nodes can be literals or field references. Replacing multiple literals can be done, however neither DynamicValidationHelper nor DynamicQueryHelper do not use this functionality. DynamicValidationHelper replaces multiple field references. DynamicQueryHelper replaces single literals/field refs.

#159 - 10/26/2017 02:59 PM - Stanislav Lomany

1. In ControlEntity, the use of LogicalTerminal.message() instead of something in ErrorManager seems incorrect. In ErrorManager we deal with things like headless mode, no-error and other session-level state. Although I think that headless mode cannot be in use in this case, I worry that the other session state may be important.

How should we handle it? I'm a bit discouraged that ErrorWriter.displayError displays error as an alert-box in GUI and as regular message in ChUI. If we have that cases, I suppose it is nice to have functions

```
ErrorWriter.displayError()
ErrorWriter.displayError(boolean alertBox)
ErrorWriter.displayError(boolean chuiAlertBox, boolean guiAlertBox)
```

#160 - 10/26/2017 05:08 PM - Greg Shah

I'm OK with adding a version that takes a boolean flag to disable the message box.

#161 - 10/27/2017 07:59 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Ovidiu, please review the changes specific to the dynamic compilation and runtime JAST interpretation code. The goal was to add dynamic conversion and interpretation for validation expressions in the same way we do for queries.

Review for 11193 of branch 3275c.
Firstly, thank you for fixing my typos. Also I see some locate() calls in DynamicQueryHelper replaced with parameters that improves a bit performance. I see the RuntimeJastInterpreter was not touch; I am glad it covers all needs of VALEXPs.
The following are just minor things I observe in the code:

- DynamicQueryHelper:550 & DynamicValidationHelper:361: the result logged is in milliseconds (divided twice with 1000 from nanos);
- DynamicQueryHelper: & DynamicValidationHelper share a lot of code. Probably we should create a class hierarchy and pull the common code up to base class (DynamicConversionHelper). This will eliminate the strange calls from one class to another;
- DynamicValidationHelper has some strange indentations, mostly when used string concatenation operations (lines: 132, 407, 416, 424, 464);
- postprocess_validation_exp.xml: copyright should be "(c) 2017". Are include and init-rules really necessary?
- BufferFieldImpl.java:1265, instead of manually prefixing the error message with ** use the prefix parameter of recordOrShowError set to true; Same for BrowseWidget.java:1072

#162 - 10/27/2017 03:41 PM - Stanislav Lomany

> - BufferFieldImpl.java:1265, instead of manually prefixing the error message with ** use the prefix parameter of recordOrShowError set to true; Same for BrowseWidget.java:1072

That's true if the error message if

```
** MESSAGE
```

but in this cases it is

```
**MESSAGE
```

(note missing space).

#163 - 10/27/2017 07:42 PM - Stanislav Lomany

Please review 3275c revision 11194, it has significant refactoring according to the notes above.

#164 - 10/28/2017 01:18 PM - Eric Faulhaber

I'm good with the persistence-related changes. Also, I successfully ran the ETF full search test suite against 3275c/11194.

#165 - 10/28/2017 09:54 PM - Greg Shah

Code Review Task Branch 3275c Revision 11194

I'm good with the changes. If it passes testing you can merge to trunk.

#166 - 10/29/2017 06:40 AM - Stanislav Lomany

Rebased task branch 3275c from P2J trunk revision 11184.

#167 - 10/30/2017 06:04 AM - Stanislav Lomany

3275c has been merged into the trunk as bzr revision 11185.

#168 - 11/09/2017 07:08 AM - Stanislav Lomany

Created task branch 3275d from trunk revision 11198 (for "column and row resizing doesn't work properly" issues which are necessary for interactive mode in enhanced browse).

#169 - 11/16/2017 07:06 AM - Stanislav Lomany

Rebased task branch 3275d from P2J trunk revision 11200. Please review.

#170 - 11/16/2017 08:21 AM - Greg Shah

Code Review Task Branch 3275d Revision 11203

The changes are good.

What testing do you recommend? The impact on ChUI seems minimal.

#171 - 11/16/2017 08:55 AM - Stanislav Lomany

Customer POC works fine, any possible regressions are related to "interactive" browse features, I think it is OK to check in.

#172 - 11/16/2017 09:04 AM - Greg Shah

Please merge 3275d to trunk.

#173 - 11/16/2017 09:54 AM - Stanislav Lomany

- Status changed from WIP to Review

3275d has been merged into the trunk as bzr revision 11201.

#174 - 11/18/2017 06:43 PM - Greg Shah

- % Done changed from 0 to 100

- Status changed from Review to Closed

#175 - 03/04/2018 08:45 AM - Greg Shah

Stanislav: I'm evaluating code for a customer application and need your feedback.

NO-VALIDATE is marked as stubs for runtime.
NO-AUTO-VALIDATE is marked as partial for runtime.

I thought we had full support for both of these. Is the gap marking wrong?

#176 - 03/04/2018 08:50 AM - Greg Shah

Stanislav: I have the same question for VIEW-AS TOGGLE-BOX. Conversion for BROWSE VIEW-AS is marked partial and runtime is marked none. What is the correct marking?

#177 - 03/04/2018 01:40 PM - Stanislav Lomany

NO-VALIDATE is marked as stubs for runtime.

It is a conversion-only option. Fully supported.

NO-AUTO-VALIDATE is marked as partial for runtime.

Fully supported.

VIEW-AS TOGGLE-BOX.

Fully supported.

VIEW-AS is marked partial and runtime is marked none

VIEW-AS COMBO-BOX is NOT supported

#178 - 03/05/2018 01:00 PM - Greg Shah

NO-VALIDATE is marked as stubs for runtime.

It is a conversion-only option. Fully supported.

OK, I've changed this.

What about the attribute?

NO-AUTO-VALIDATE is marked as partial for runtime.

Fully supported.

Changed.

VIEW-AS TOGGLE-BOX.

Fully supported.

The runtime works too, right?

VIEW-AS is marked partial and runtime is marked none

VIEW-AS COMBO-BOX is NOT supported

I plan to mark both conversion and runtime as partial.

#179 - 03/05/2018 03:08 PM - Stanislav Lomany

> What about the NO-VALIDATE attribute?

Full support too.

> The runtime (VIEW-AS TOGGLE-BOX) works too, right?

Yes.

Files

row-display.png	16.8 KB	05/25/2017	Stanislav Lomany
brws-pgdown.png	11.5 KB	06/06/2017	Stanislav Lomany
auto-return-br-bug.png	4.01 KB	08/07/2017	Stanislav Lomany
brws-auto-return.p	729 Bytes	08/07/2017	Stanislav Lomany
toggle-box.png	3.49 KB	08/21/2017	Stanislav Lomany