User Interface - Feature #3281

implement some frame options

04/27/2017 02:15 PM - Greg Shah

Status:	Closed	**Start date:**	04/27/2017
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**version:**	
vendor_id:	GCD		

Description

Related issues:

Related to Database - Feature #2692: table-level VALEXP/VALMSG support	**Closed**
Related to Bugs - Bug #3347: HELP string support doesn't work for demo_widgets.p	**WIP**

History

#1 - 04/27/2017 02:16 PM - Greg Shah

Implement the following frame options (see #3257-14):

NO-VALIDATE
USE-DICT-EXPS
NO-ATTR-SPACE
COLOR PHRASE (the app uses "blu/brown", "White/blu", col-norm and col-input)

#2 - 08/14/2017 04:41 PM - Constantin Asofiei

Some initial investigation:

1. NO-VALIDATE - looks like this is already present at conversion time
2. USE-DICT-EXPS - conversion time only, must enable db-level validation expressions for all fields in the frame. some testing is required to see how deep does the 4GL compiler go to activate this: what if the option is present before a frame:sensitive = true call?
3. NO-ATTR-SPACE - conversion time only; although ATTR-SPACE and NO-ATTR-SPACE can't co-exist in the same frame options phrase, they can exist in different frame options:

```
form ch i with frame f3c side-labels attr-space  title "f3c".
update ch i with frame f3c no-attr-space.
```

   FWD currently takes attr-space as set. Need to confirm how 4GL behaves - does it take last setting or first on?
4. COLOR PHRASE - there are some conversion issues; runtime looks like is implemented, but it needs to be double-checked, especially for the col-norm and col-input

#3 - 08/14/2017 05:08 PM - Greg Shah

> NO-VALIDATE - looks like this is already present at conversion time

I don't see the **option** being handled. There is a browse option of the same keyword but it is commented out. I see only an attribute being supported for browse.

I also don't see the runtime stubs for any option.

It is possible there are no runtime implications. If so, then we would at least need to honor the option at conversion time (by avoiding schema validations when calculating the validation expressions).

> some testing is required to see how deep does the 4GL compiler go to activate this: what if the option is present before a frame:sensitive = true call?

My guess is that it does not check assignments to SENSITIVE at all. The rvalue can be runtime-calculated so it is not very useful. In other words, I think this frame option causes all fields to have schema validations if there isn't already an override validation expression previously specified. If ENABLE or SENSITIVE are not ever used for a field, then the validation expression is dead code.

#4 - 08/14/2017 05:11 PM - Greg Shah

- Related to Feature #2692: table-level VALEXP/VALMSG support added

#5 - 08/14/2017 05:30 PM - Constantin Asofiei

Greg Shah wrote:

> > NO-VALIDATE - looks like this is already present at conversion time
>
> I don't see the **option** being handled. There is a browse option of the same keyword but it is commented out. I see only an attribute being supported for browse.

See this code in annotations/validation_prep.rules:156:

```
        <!-- detect the presense of NO-VALIDATE option in this frame -->
        <action>framenov = downPath("FRAME_PHRASE/KW_NO_VALID")</action>
```

This disables the emit of any validation expressions.

> I also don't see the runtime stubs for any option.
>
> It is possible there are no runtime implications. If so, then we would at least need to honor the option at conversion time (by avoiding schema validations when calculating the validation expressions).

Yes, I don't think there is any runtime code needed for this - just conversion, and the rules seems to be in place (as confirmed by a simple test).

> > some testing is required to see how deep does the 4GL compiler go to activate this: what if the option is present before a frame:sensitive = true call?
>
> My guess is that it does not check assignments to SENSITIVE at all. The rvalue can be runtime-calculated so it is not very useful. In other words, I think this frame option causes all fields to have schema validations if there isn't already an override validation expression previously specified. If ENABLE or SENSITIVE are not ever used for a field, then the validation expression is dead code.

Yes, SENSITIVE is not checked; 4GL compiler looks like it picks up this option regardless where it is set, in a pre-parsing phase (as is picked up even if is in an unreachable code). Currently we attach validation expressions only if there is an ENABLE/UPDATE/SET/PROMPT-FOR statement for such a field, at the parser level; for USE-DICT-EXPS we need to attach the validation expressions and also HELP (forgot to mention this) for all fields in the frame, I think sometime in the post-parsing fixups phase.

#6 - 08/15/2017 08:47 AM - Constantin Asofiei

The USE-DICT-EXPS stmt is interesting:

1. any VALIDATE or HELP options set at the widget, before USE-DICT-EXPS was set (including current frame statement), have priority over the schema help/validate
2. any VALIDATE or HELP options set at the widget, after USE-DICT-EXPS was set, will be ignored.

This might be consistent with the top-down parsing: 4GL collects HELP/VALIDATE widget options and once USE-DICT-EXPS is encountered, it switches the help/validate collection mode to include only schema values, but it does not clear the already collected widget options.

Not sure yet how it behaves when NO-VALIDATE is in play: I've already found a case which works incorrectly in FWD (NO-VALIDATE enforces "first frame reference setting" only for the first statement which is not FRAME or DEFINE FRAME).

#7 - 08/15/2017 10:05 AM - Greg Shah

NO-VALIDATE - looks like this is already present at conversion time

I don't see the **option** being handled. There is a browse option of the same keyword but it is commented out. I see only an attribute being supported for browse.

See this code in annotations/validation_prep.rules:156:
[...]
This disables the emit of any validation expressions.

Good catch, I missed it. Please put comments in the frame generator that explain that the kw_no_valid processing is in annotations/validation_prep.rules and does not need to be treated as a settable value in the frame.

I also don't see the runtime stubs for any option.

It is possible there are no runtime implications. If so, then we would at least need to honor the option at conversion time (by avoiding schema validations when calculating the validation expressions).

Yes, I don't think there is any runtime code needed for this - just conversion, and the rules seems to be in place (as confirmed by a simple test).

Please change the runtime marking to RT_LVL_FULL_RESTR and put a comment on that line that explains that "no runtime support is needed (not even stubs), all support is implemented during conversion (no setters are called on the frame widget)".

I do wonder if there is some implication when we use CREATE FRAME that would require the option to be known at runtime.

#8 - 08/15/2017 02:08 PM - Constantin Asofiei

Another validation issue: the schema-level validation is pulled for a field on its first occurrence in a UPDATE/ENABLE/SET/PROMPT-FOR statement; for example:

```
update book.cost with frame f1.
update book.cost validate(book.cost > 2, "> 2") with frame f1.
```

will use the schema-level validation for book.cost, as the explicit validate phrase appears later in the code. FWD is not working properly here.

#9 - 08/15/2017 03:52 PM - Constantin Asofiei

USE-DICT-EXPS behaviour in 4GL looks like is just a way of enforcing the schema-level validation (and help) for the field widgets referenced in any UI statement, not just UPDATE/ENABLE/etc. Once this mode is seen as set for the frame, first encounter for the field will pull the schema-level validation/help UNLESS there is an explicit one set in the current or a previous statement.

Greg, please confirm this: current approach in FWD is to let the parser emit schema-level validation expressions for the fields and, if there are multiple validation expressions, conversion rules will take care of keeping only the first encountered validation expression. But I'm having trouble finding where this filtering occurs in the TRPL rules.

What I'm planning is to modify the parser to track USE-DICT-EXPS option for frames and let the schema validation (and help) emit for any statement, once it was set.

#10 - 08/15/2017 04:26 PM - Greg Shah

#2692-36 has a summary of the most recent work on implicit validation processing. Stanislav also has changes in branch 2892a that implement validation processing for browse (see #3275).

The key thing is that we now process validation expressions in the parser instead of in the schema dictionary. By matching the Progress compiler's timing/scope for when to parse the validation expressions we can now handle the local procedure resource/code references that can be present in the schema. To handle the implicit ENABLE ALL case, we track the widgets in each frame (see sym.addFrameFields() and sym.getFrameFields() in the parser).

> current approach in FWD is to let the parser emit schema-level validation expressions for the fields and, if there are multiple validation expressions, conversion rules will take care of keeping only the first encountered validation expression. But I'm having trouble finding where this filtering occurs in the TRPL rules.

Sort of. In annotations/validate_prep.rules we mark invalid validation expressions with an "ignore" annotation. Then in annotations/validation.rules we bypass the processing of these "ignore" expressions and only process the ones that are not marked "ignore". In addition, in annotations/validation_post.rules, we track the first use and hide those validation expressions that are not first. From there, in annotations/validation_post2.rules we only copy the non-hidden ones into the VALIDATION node from which we will emit the converted result later.

So there is not really a delete, but instead we drop the extra references in stages later using annotations and hiding.

> What I'm planning is to modify the parser to track USE-DICT-EXPS option for frames and let the schema validation (and help) emit for any statement, once it was set.

OK, this seems reasonable.

Eric: do you have any comments?

#11 - 08/15/2017 04:53 PM - Constantin Asofiei

Greg, thanks for the explanation - it will help.

> What I'm planning is to modify the parser to track USE-DICT-EXPS option for frames and let the schema validation (and help) emit for any statement, once it was set.

> OK, this seems reasonable.
>
> Eric: do you have any comments?

I have a comment: I can't really track per-frame USE-DICT-EXPS at the parser level without having logic related to frame scoping to resolve the frames correctly... in FWD this is done sometime at the annotations phase; plus, I can't pull the schema-level validation expression after the fixups phase, as we will miss any processing done (for buffers and other stuff). So, what about this change in direction: the schema level validation/help is pulled by any UI statement (not just ENABLE/UPDATE/etc), and TRPL rules will later take care of 'ignoring' the ones which are not needed (i.e. if USE-DICT-EXPS is not set for the frame). I'll be able to track safely this flag in the annotation phase, in validation_prep.rules and adjust/ignore the validation/help accordingly, while the validation expression will be fully processed by our TRPL rules.

#12 - 08/15/2017 05:00 PM - Greg Shah

The SymbolResolver does track per-frame data. While its processing may not correspond 100% to the frame scoping rules, it does use block level scoping to handle this. See references to fieldWidgetDict.

I think it is best to use this same approach to track the USE-DICT-EXPS per frame. Keep a separate SSD which has an entry for those frames in which the USE-DICT-EXPS has been set.

The result will be much less change than trying to clean up later. And any scoping problem is no worse than that which we already have.

#13 - 08/15/2017 05:23 PM - Eric Faulhaber

Greg Shah wrote:

> Eric: do you have any comments?

Nothing to add to your summary; that is pretty much how I recall it.

Constantin, when working on my part of the solution for ENABLE ALL, I wasn't aware of these other requirements, but based on my limited

understanding of USE-DICT_EXPS (only from the docs and your description above), it seems we can use the approach Greg suggests, unless you find the scope information available in the SymbolResolver somehow insufficient.

#14 - 08/16/2017 07:36 AM - Constantin Asofiei

Greg, please review 3260a rev 11160 - it includes all issues in note 1, except the schema help which can be pulled by USE-DICT-EXPS - I haven't added this yet.

#15 - 08/16/2017 04:29 PM - Constantin Asofiei

- % Done changed from 0 to 70

- Status changed from New to WIP

- Assignee set to Constantin Asofiei

3260a 11161 adds the HELP for USE-DICT-EXPS. Please review. Only the COLOR runtime remains to be implemented.

#16 - 08/17/2017 04:53 PM - Constantin Asofiei

There is an issue with USE-DICT-EXPS with shared frames and HELP: this setting is local to the program where the shared frame is being used. For VALIDATE, we emit code inside the converted Java program. I think we need something similar for the HELP option.

Not sure how fast I can solve it.

#17 - 08/17/2017 04:54 PM - Greg Shah

> Not sure how fast I can solve it.

Solve it later. Focus on the other stuff that needs to happen for the customer deadline.

#18 - 08/17/2017 04:57 PM - Constantin Asofiei

Greg Shah wrote:

> > Not sure how fast I can solve it.
>
> Solve it later. Focus on the other stuff that needs to happen for the customer deadline.

OK, then I'm disabling all the HELP-related code for USE-DICT-EXPS, as it causes regressions in the ChUI regression tests.

#19 - 08/17/2017 05:06 PM - Greg Shah

> then I'm disabling all the HELP-related code for USE-DICT-EXPS, as it causes regressions in the ChUI regression tests

OK.

#20 - 09/08/2017 04:08 PM - Ovidiu Maxiniuc

While working on #3330 I identified a conversion issue: if normally in a CAN-FIND (or other query) an field without without table prefix is associated with the table/buffer of the query, in the case of validation, it should be associated with the field that is defined.

For example, consider this piece of data definition:

```
ADD TABLE "author"   DUMP-NAME "author"
ADD FIELD "author-id" OF "author"  AS integer   ORDER 10

ADD TABLE "book"  DUMP-NAME "book"
ADD FIELD "author-id" OF "book" AS integer
  VALEXP "CAN-FIND(author WHERE author.author-id = author-id)"
  VALMSG "Author not found"
  ORDER 10
```

and the procedure:

```
DEFINE FRAME fr0
    book.author-id AT ROW 1 COL 1
    WITH 3 DOWN TITLE "Books".

CREATE book.
DISPLAY book.author-id WITH FRAME fr0.
ENABLE ALL WITH FRAME fr0.
```

When converting, FWD will see author.author-id = author-id in a CAN-FIND(author...) so it will assume this is a tautology and drop it. As result, the generated validation will look like this:

```
fr0Frame.widgetAuthorId().setValidation(((ValidationExpr<integer>) (integer authorId_) ->
         new FindQuery(author, (String) null, null, "author.id asc", LockType.NONE).hasOne()), "Author not fou
nd");
```

which is the equivalent of "there is at least one author in database".

#21 - 09/11/2017 02:16 PM - Ovidiu Maxiniuc

I've done some investigations related to the issue from note 20. The decision where the unqualified field belongs to happens very early, in parsing stage. The stack looks like this:

```
at com.goldencode.p2j.schema.SchemaDictionary.getFieldInfo(SchemaDictionary.java:1565)
at com.goldencode.p2j.uast.SymbolResolver.lambda$lookupFieldInfo$23(SymbolResolver.java:4356)
at com.goldencode.p2j.uast.SymbolResolver.lambda$processHierarchy$35(SymbolResolver.java:6044)
at com.goldencode.p2j.uast.SymbolResolver.processHierarchy(SymbolResolver.java:6103)
at com.goldencode.p2j.uast.SymbolResolver.lookupFieldInfo(SymbolResolver.java:4364)
at com.goldencode.p2j.uast.SymbolResolver.annotateField(SymbolResolver.java:1608)
at com.goldencode.p2j.uast.ProgressParser.lvalue(ProgressParser.java:14654)
[...]
at com.goldencode.p2j.uast.ProgressParser.can_find_function(ProgressParser.java:55288)
[...]
at com.goldencode.p2j.uast.ProgressParser.expr(ProgressParser.java:8933)
at com.goldencode.p2j.uast.ProgressParser.attachSchemaValidation(ProgressParser.java:918)
at com.goldencode.p2j.uast.ProgressParser.attachAssignSchemaValidation(ProgressParser.java:759)
at com.goldencode.p2j.uast.ProgressParser.attachAssignSchemaValidation(ProgressParser.java:724)
at com.goldencode.p2j.uast.ProgressParser.format_phrase(ProgressParser.java:18088)
[...]
at com.goldencode.p2j.uast.AstGenerator.parse(AstGenerator.java:1487)
at com.goldencode.p2j.uast.AstGenerator.processFile(AstGenerator.java:957)
at com.goldencode.p2j.uast.ScanDriver.lambda$scan$0(ScanDriver.java:375)
```

The SchemaDictionary.getFieldInfo cannot identify the correct table for the unqualified field author because the can_find_function inserts a stronger scope with p2j_test.author.author-id as preferred node and promoting p2j_test.author. The correct table can be found in the SchemaDictionary.scopes one level up (in this case), where the promoted table is p2j_test.book.

I am not sure if a simple solution exist here. We need to introduce some exception when computing the parent table of an unqualified field in the case of the validation: the top priority must be filed whose validation is attached from the schema. This breaks the natural traversal of the dictionary.

#22 - 09/14/2017 11:44 AM - Greg Shah

> We need to introduce some exception when computing the parent table of an unqualified field in the case of the validation: the top priority must be filed whose validation is attached from the schema. This breaks the natural traversal of the dictionary.

This idea seems reasonable. However, I don't think we can promote at the field level. Promotion is only a table concept right now.

On the other hand, we certainly can put special logic in the validation processing of the parser. In fact, we already have code that is aware of the validation flag in resolveLvalueCoreType() which is where we do the lookup for a field match of an lvalue in an expression. So some code there could be changed to deal with this case.

The validate rule is called from format_phrase, column_ref and delete_stmt. In format_phrase we already track the validateFieldName which is the schemaname of the lvalue if it is a field. If null, then there is no special processing needed. But if it is non-null, then that is the "precedence match" that would need to be applied in resolveLvalueCoreType().

The column_ref will have to be changed to track validateFieldName similar to format_phrase. Having validateFieldName as null for delete_stmt is already correct so no changes there.

Do we need to do some kind of special schema dictionary lookup and combine that with the validate and validateFieldName to properly resolve the field? I don't want to duplicate any of the schema dictionary parsing in resolveLvalueCoreType(), so I suspect it may be needed.

#23 - 09/14/2017 11:45 AM - Greg Shah

Ovidiu: Are you working this one?

#24 - 09/14/2017 11:56 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> Ovidiu: Are you working this one?

Yes, I am using the testcase to go step-by-step and try to find a door where we can change the priorities. I have two ideas at this moment:

1. use a ProgresParser member field to store the field when attaching schema validators. It should contain the name of the database field whose validation is processed and resolve it in a dedicated code. validateFieldName already exist but used for different purpose and I am not sure this is enough;
2. using a priority promotion in dictionary. The validation will be using a higher level so that it will be preferred to can-find. This is more complicated.

#25 - 09/14/2017 12:10 PM - Greg Shah

> use a ProgresParser member field to store the field when attaching schema validators. It should contain the name of the database field whose validation is processed and resolve it in a dedicated code. validateFieldName already exist but used for different purpose and I am not sure this is enough;

The purpose of validateFieldName should be exactly what is needed here. The field resolution fails in resolveLvalueCoreType(). In the case of attaching schema validators, then why not use the validateFieldName, since it won't be otherwise set except for in format_phrase (and column_ref when you fix it). The attachment of schema validators is done outside of there, right?

> using a priority promotion in dictionary. The validation will be using a higher level so that it will be preferred to can-find. This is more complicated.

How about if the SchemaDictionary provided a worker method to compare the candidate field name to see if it was a match to the validateFieldName (which is a fully qualified schema name)? The code in resolveLvalueCoreType() can call this if validation == true and validateFieldName != null. The idea is that the core promotion logic is not changed. But we still leverage the schema dictionary's correct handling of all the namespace/qualifiers/abbreviations/case-insensitivity processing.

#26 - 09/14/2017 12:11 PM - Greg Shah

Eric: Do you have any guidance here?

#27 - 09/14/2017 12:18 PM - Eric Faulhaber

Greg Shah wrote:

> How about if the SchemaDictionary provided a worker method to compare the candidate field name to see if it was a match to the validateFieldName (which is a fully qualified schema name)?

What would be the criteria for determining the match?

#28 - 09/14/2017 12:30 PM - Greg Shah

I think the method could be something like boolean SchemaDictionary.specificFieldMatch(String candidate, String schemaname).

If candidate is unqualified, then it should be checked to see if it is a match to the field portion of the fully qualified schemaname. I think qualified names don't have to match here.

Ovidiu: do you see any other logic needed?

#29 - 09/14/2017 01:33 PM - Eric Faulhaber

Maybe I'm missing something, but it seems like it is more complicated than that. Doesn't such a match need to consider and handle abbreviations?

The abbreviation algorithm is encapsulated in the find logic of the Namespace class. It was not written to support standalone comparisons against single names, but rather to find the best match from a pool of candidate names. By its nature, matching abbreviated names depends on having such a pool, so it can determine whether there's a match at all, and whether there's ambiguity among multiple, possible matches.

#30 - 09/14/2017 02:18 PM - Greg Shah

> Doesn't such a match need to consider and handle abbreviations?

Yes. It should handle abbreviations and case-insensitivity.

> By its nature, matching abbreviated names depends on having such a pool, so it can determine whether there's a match at all, and whether there's ambiguity among multiple, possible matches.

We are just getting a yes/no answer here. It either matches this name or it doesn't. In this case, the pool of abbreviations is limited to the specific unqualified field name being processed. We don't need the normal namespace processing.

#31 - 09/14/2017 04:31 PM - Ovidiu Maxiniuc

Greg, I have a fix for this issue. It works fine for my testcases. Shall I commit it to 3330a for review?

#32 - 09/14/2017 04:33 PM - Greg Shah

Yes, please do.

#33 - 09/14/2017 04:42 PM - Ovidiu Maxiniuc

Committed as revision 11237.

#34 - 09/14/2017 05:02 PM - Greg Shah

Code Review Task Branch 3330a Revision 11237

Eric: please review and comment.

1. My sense is that this is encoding a bit too much knowledge of the specific use case (validation) into the schema dictionary. Previous APIs exposed were designed to be more generic.

2. In SchemaDictionary.setValidateField() I think that fdp == ldp may be valid for temp-table cases. I thought that the schemaname in that case only has a single .. In addition, this code is manually parsing the schemaname portions, which is normally delegated to EntityName. This seems wrong because it duplicates processing that is best left in a single location.

3. Even SchemaDictionary.checkValidateField() is basically doing some manual processing of case and abbreviations. I don't think we need this code in SchemaDictionary if we are just going to manually calculate things. And the manual calculation may be the right thing to do (unless Eric sees a better way using existing features). But if this manual calculation is to be done, then we might as well keep this code in the parser (in resolveLvalueCoreType()).

4. The current approach only handles the case during attachment of the schema validations. I think the same problem exists during the call to validate from both format_phrase and from column_ref.

#35 - 09/14/2017 05:18 PM - Greg Shah

To be clear about this, I am proposing that all the processing for this be removed from SchemaDictionary (and SymbolResolver) and instead it should be included in the parser. It can then be called from resolveLvalueCoreType().

In addition to handling the setting of validateFieldName and the validation flag in the schema attaching code, also handle it in column_ref. Then the resolveLvalueCoreType() processing (or the code it calls) will be unified. I don't see any reason why any state is needed except for validateFieldName and the validation flag.

The processing can be close to the current approach, although the splitting of the qualified name is probably better to be done by reusing EntityName.

#36 - 09/14/2017 05:19 PM - Eric Faulhaber

Greg Shah wrote:

> Code Review Task Branch 3330a Revision 11237
>
> Eric: please review and comment.

Greg, I agree with all the points you raised in your initial review. This new code seems out of place in SchemaDictionary; it doesn't need or use any of the state in that class, but instead overrides the lookup the schema dictionary normally would do, based on a particular parser state. It seems better placed in the parser as you suggest.

#37 - 09/14/2017 05:21 PM - Ovidiu Maxiniuc

I understand. I will adjust the code as Greg suggested.

#38 - 09/15/2017 02:05 PM - Ovidiu Maxiniuc

I moved the code into Progress parser. In fact I used the already existing code from resolveLvalueCoreType.
Support for column_ref (and delete_stmt) was added through validate() which now has as parameter the schemaname of processed field.

Committed as r11240. Please review.

#39 - 09/15/2017 02:16 PM - Constantin Asofiei

Ovidiu, while you are at it, there is another case where I think the condition is dropped: if the VALEXP "not CAN-FIND(book WHERE book.author-id = author-id)" - in this case, the r-value author-id, should it be assumed as the INPUT author-id or book.author-id field?

#40 - 09/15/2017 02:40 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, while you are at it, there is another case where I think the condition is dropped: if the VALEXP "not CAN-FIND(book WHERE book.author-id = author-id)" - in this case, the r-value author-id, should it be assumed as the INPUT author-id or book.author-id field?

I assume that the valexped field is "author-id" of "book". It is the validated field. The generated code looks like this:

```
fr0Frame.widgetAuthorId().setValidation(((ValidationExpr<integer>) (integer authorId_) -> not(new FindQuery(au
thor, "author.authorId = ?", null, "author.id asc", new Object[]
{
   authorId_
}, LockType.NONE).hasOne())), "Author not found");
```

This is INPUT author-id, in fact, because the ValidationExpr is set on fr0Frame.widgetAuthorId(). If you write it explicitly like INPUT author-id, the code will change so that the client-side parameter for query will be

```
(P2JQuery.Parameter) () -> fr0Frame.getAuthorId()
```

which should provide the same value like authorId_.

On the other side, writing explicitly book.author will generate no difference from the first code since now author-id is correctly expanded to fully qualified p2j_test.book.author-id, the same as schema name of book.author.

Why should have it been dropped?

#41 - 09/15/2017 02:57 PM - Greg Shah

Code Review Task branch 3330a Revision 11240

This is very close.

1. I think the unconditional use of (String) #l.getAnnotation("schemaname") in column_ref is a problem if the lvalue is not a field. I think that case is possible. It should be protected as is done in format_phrase.

2. I don't think that delete_stmt needs to pass the schemaname. The delete_stmt operates on a record, not a field. We just reuse the validate rule there because the structure is the same. But I don't think any validation expression processing in that location is actually giving any special precedence to a specific field. Even if it did, we are passing the schemaname of the record, which is a fully qualified table name. I think delete_stmt should just pass null.

#42 - 09/15/2017 03:10 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Why should have it been dropped?

I meant FWD previously dropped it, not that it should.

#43 - 09/15/2017 04:24 PM - Ovidiu Maxiniuc

Greg,
You are right. I updated the code accordingly. Committed as r11241.

#44 - 10/04/2017 06:01 PM - Constantin Asofiei

Greg Shah wrote:

> COLOR PHRASE (the app uses "blu/brown", "White/blu", col-norm and col-input)

In the code I see only value(col-norm) or value(col-input) - is this what you are referring to? Because there is no predefined col-norm or col-input in 4GL.

#45 - 10/05/2017 10:08 AM - Greg Shah

I think "blu/brown" and "White/blu" were dropped out of the code when we removed from files as out of scope.

> In the code I see only value(col-norm) or value(col-input) - is this what you are referring to?

Yes.

#46 - 10/06/2017 04:51 PM - Constantin Asofiei

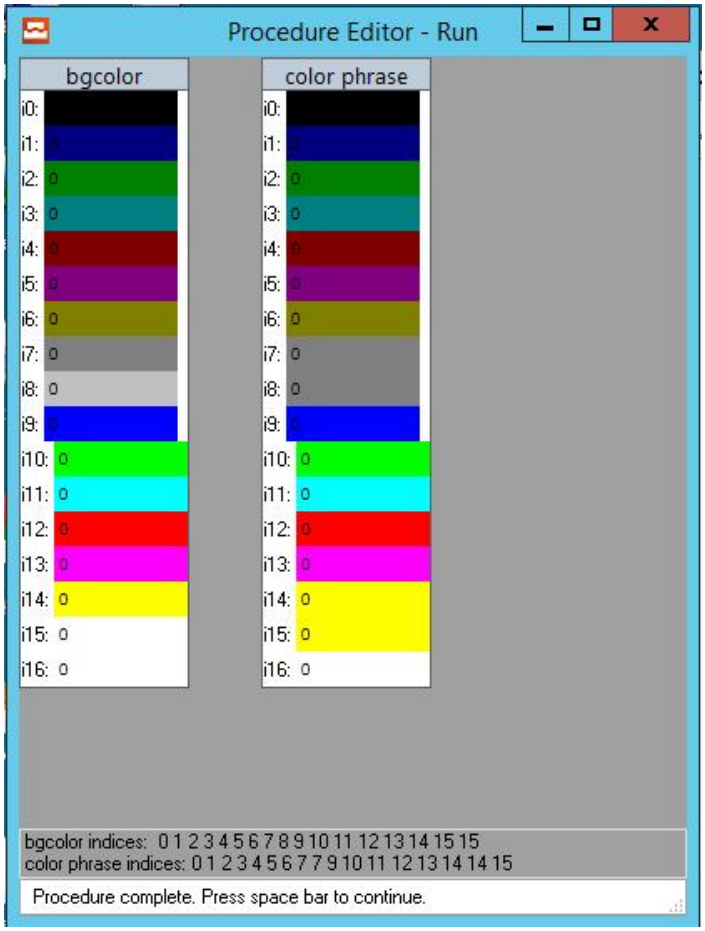
3281a rev 11175 should fix the issue with the HELP and USE-DICT-EXPS; running testing now.

#47 - 10/06/2017 06:12 PM - Constantin Asofiei

- File color_phrase.png added

So, when COLOR statement uses color names in GUI, there is no RGB or other stuff like this computed... each color name has an associated index (from 0 to 15), from the color table; whatever RGB definition is there, that will be used (thus you can end up with White color name drawn as red).

See attached picture for how BGCOLOR attribute and COLOR statement is used for color names as in the https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dvref/color-phrase.html table; the message area contains the color indices for BGCOLOR (explicitly set) and for the color names:



I need to test the color names with prefixes like BLINK-, RVV-, UNDERLINE- and BRIGHT-, and see how the index in the color table gets computed. The good news is we don't have to implement anything on client-side, the server-side just needs to set the BGCOLOR and FGCOLOR attributes accordingly, to a color index in the 0 to 15 range.

#48 - 10/06/2017 07:20 PM - Constantin Asofiei

Greg, I need some help; the regression related to HELP is about ENABLE ALL: enable_stmt rule wants to iterate all frame fields, and if a field has a VALEXP or HELP which needs to be pulled from the schema, then it adds a new AST (under KW_ALL) with the field and a format phrase with the valexp/help... this is not correct, as KW_ALL:

1. will lose any EXCEPT fields... information
2. downstream processing will no longer be able to convert KW_ALL properly

So, any idea how to do this differently?

#49 - 10/09/2017 07:35 AM - Greg Shah

In regard to the color names, I assume you are talking about the following (from Progress docs):

- Black, Bla, Blk
- Blue, Blu
- Green, Gre, Grn
- Cyan, C
- Red
- Magenta, Ma
- Brown, Bro, Brn
- Gray, Gra, Gry (what about Grey, Gre?)
- Dark-Gray, D-Gra (what about Dark-Grey, D-Gre?)
- Light-Blue, Lt-Blu
- Light-Green, Lt-Gre (what about Lt-Grn?)
- Light-Cyan, Lt-C
- Light-Red, Lt-Red
- Light-Magenta, Lt-Ma
- Light-Brown, Lt-Bro (what about Lt-Brn?)
- Yellow, Y
- White, W

Have you tried to see if arbitrary abbreviations work? For example, are Wh, Whi, Whit all the same as White?

What does Progress do if you pass something other than one of these "well known" names?

#50 - 10/09/2017 07:37 AM - Constantin Asofiei

Greg Shah wrote:

> In regard to the color names, I assume you are talking about the following (from Progress docs):

Yes.

> Have you tried to see if arbitrary abbreviations work? For example, are Wh, Whi, Whit all the same as White?

Not yet.

> What does Progress do if you pass something other than one of these "well known" names?

I think it ignores the setting, but I will double check.

#51 - 10/09/2017 07:38 AM - Constantin Asofiei

Constantin Asofiei wrote:

> Greg, I need some help; the regression related to HELP is about ENABLE ALL: enable_stmt rule wants to iterate all frame fields, and if a field has a VALEXP or HELP which needs to be pulled from the schema, then it adds a new AST (under KW_ALL) with the field and a format phrase with the valexp/help... this is not correct, as KW_ALL:
>
> 1. will lose any EXCEPT fields... information
> 2. downstream processing will no longer be able to convert KW_ALL properly
>
> So, any idea how to do this differently?

Regarding this: we can't solve ENABLE ALL [EXCEPT] at parser time, as we don't have the full frame definition at the point this statement is reached; I need to mark this to either keep or discard schema validation and help, so it will be resolved at conversion time.

#52 - 10/09/2017 12:00 PM - Eric Faulhaber

Constantin Asofiei wrote:

> Greg, I need some help; the regression related to HELP is about ENABLE ALL: enable_stmt rule wants to iterate all frame fields, and if a field has a VALEXP or HELP which needs to be pulled from the schema, then it adds a new AST (under KW_ALL) with the field and a format phrase with the valexp/help... this is not correct, as KW_ALL:
>
> 1. will lose any EXCEPT fields... information
> 2. downstream processing will no longer be able to convert KW_ALL properly

> So, any idea how to do this differently?

Constantin, I wrote the original code to append the schema validation nodes, but it looks like it has been changed since then, and I am not up to speed on the problem you are dealing with now. Please help me understand the issue you are reporting. Specifically...

> will lose any EXCEPT fields... information

Where was this information and at what point is it being lost? It would help if you could provide some snippets of example AST to refresh my memory of the correct structure and what the damaged structure looks like. Or is your point that we need to figure out what the correct structure should be?

> downstream processing will no longer be able to convert KW_ALL properly

What is going wrong downstream, exactly, and where in the conversion flow?

#53 - 10/09/2017 12:13 PM - Constantin Asofiei

Eric Faulhaber wrote:

> Constantin, I wrote the original code to append the schema validation nodes, but it looks like it has been changed since then, and I am not up to speed on the problem you are dealing with now.

First let me give you some notes; enable_stmt was using (in case of KW_ALL) a fields = sym.getFrameFields(frame); call to get the frame's fields - but this is not correct, as code can add field widgets to the frame AFTER this ENABLE ALL statement; so the assumption that all fields will be processed here is incorrect, as at parser time there is not enough knowledge about the frame definition to make a decision like 'these are all the field widgets in this frame, so lets pull validation expressions for them'.

> > will lose any EXCEPT fields... information
>
> Where was this information and at what point is it being lost? It would help if you could provide some snippets of example AST to refresh my memory of the correct structure and what the damaged structure looks like. Or is your point that we need to figure out what the correct structure should be?
>
> > downstream processing will no longer be able to convert KW_ALL properly
>
> What is going wrong downstream, exactly, and where in the conversion flow?

Considering the point above that parser can't decide to pull schema valexp or help for ENABLE ALL, I haven't investigated further; but basically, in case of ENABLE ALL EXCEPT, the AST looks like:

```
KW_ALL
   KW_EXCEPT
   CONTENT_ARRAY
     field1
     field2
     ...
     fieldn
```

In the original code, when this was executed:

```
                    astFactory.addASTChild(allPair, wid);
                    astFactory.addASTChild(allPair, fmt);
```

the KW_ALL had its KW_EXCEPT and CONTENT_ARRAY children removed and replaced by the fields which had the valexp pulled from schema.

So, I have a solution at least for schema HELP, in 3281a rev 11176, but I haven't checked yet if it fixes VALEXP too. Are there any existing tests for VALEXP and ENABLE ALL?

#54 - 10/09/2017 01:29 PM - Eric Faulhaber

Constantin Asofiei wrote:

> Are there any existing tests for VALEXP and ENABLE ALL?

Not very sophisticated ones; I was just trying to solve a very specific problem at the time. See val-exp*.p. They rely on the p2j.book table having validation expressions (in the cost and price fields).

> Considering the point above that parser can't decide to pull schema valexp or help for ENABLE ALL...

Do we need to move the AST augmentation we're doing for schema validation and help expressions from the parser to a worker that gets invoked further downstream (i.e., after the frame fields have been fully determined)? Greg, do you recall why we didn't do that initially? Was there some TRPL processing that had to happen pretty early on, such that we decided to put this in the parser in the first place?

#55 - 10/09/2017 01:32 PM - Constantin Asofiei

Eric Faulhaber wrote:

> Do we need to move the AST augmentation we're doing for schema validation and help expressions from the parser to a worker that gets invoked further downstream (i.e., after the frame fields have been fully determined)?

Well, with my changes SymbolResolver.getFrameFields is no longer used by the parser; and the only usage was the case of ENABLE ALL. The other work done by the parser related to schema valexp and help is OK, as it will work with explicitly specified field widgets.

#56 - 10/09/2017 02:00 PM - Greg Shah

> Greg, do you recall why we didn't do that initially? Was there some TRPL processing that had to happen pretty early on, such that we decided to put this in the parser in the first place?

The way I remember it is that the statement location was important. I know the statement location was needed for parsing the state references (e.g. variables) that could exist in the business logic. By parsing the valexp later during fixups or annotations (after the program's symbols and other resources were gone), the parsing would fail. In #2692, it was needed to move these into the parser. Moving the valexp parsing to a later time wpould require us to duplicate the variable dictionary, function dictionary and other state of the business logic which is something we really want to avoid doing.

I also remember that the different sets of fields would get schema validation depending on where the ENABLE ALL statement existed in the code. I could be wrong about this.

Constantin: Are you sure that widgets added later in the source file are definitely included when they are only referenced by a previous ENABLE ALL (and no other SET, PROMPT-FOR, UPDATE, INSERT or ENABLE)? I thought that had been checked.

As far as I know, Progress never had a multi-pass feature of the compiler until the OO stuff. I really doubt this validation stuff (which is really old) had anything so sophisticated. Unless it was done at runtime, but I think we have already found that in an r-code approach, the Progress dynamic compilation of expressions is quite limited (see #3275 regarding the dynamic validation expressions for browse).

#57 - 10/09/2017 02:08 PM - Constantin Asofiei

Greg Shah wrote:

> I also remember that the different sets of fields would get schema validation depending on where the ENABLE ALL statement existed in the code. I could be wrong about this.

Greg, thanks, this was the key question for this issue: the 4GL runtime for ENABLE ALL does see all widgets in the frame (no matter where they are defined), but it doesn't pull HELP (and I assume VALEXP) for widgets defined later in the code! See this test, assuming book.book-title has schema-level HELP:

```
def var i1 as int.
def var i2 as int.
find first book.

enable all with frame f1.

wait-for go of frame f1. /* no help seen for book.book-title */

display book.book-title i1 i2 with frame f1.
```

So the parser code added by Eric in enable_stmt was OK, as it was looking into only the frame fields defined at this point; I need to backout my changes and see why/how that code no longer works.

#58 - 10/09/2017 03:09 PM - Greg Shah

If the EXCEPT list is destroyed by the addition of schema validations, it should be fixed. Do we honor it properly?

#59 - 10/09/2017 03:13 PM - Constantin Asofiei

Greg Shah wrote:

> If the EXCEPT list is destroyed by the addition of schema validations, it should be fixed. Do we honor it properly?

Yes, all KW_ALL children are lost. Please take a look at this code in enable_stmt (at the end of the rule):

```
               ASTPair allPair = new ASTPair();
               astFactory.makeASTRoot(allPair, #a);
               for (Aast fld : fields)
               {
                  String sname = (String) fld.getAnnotation("schemaname");
                  Aast wid = sym.createField(sname);
```

```
                wid.putAnnotation("enable_all", true);
                Aast fmt = new ProgressAst();
                ASTPair fmtPair = new ASTPair();
                astFactory.makeASTRoot(fmtPair, fmt);
                List<Aast> asts = attachAssignSchemaValidation(fmtPair, wid, sym);
                if (asts != null && !asts.isEmpty())
                {
                   astFactory.addASTChild(allPair, wid);
                   astFactory.addASTChild(allPair, fmt);
                   wid.putAnnotation("enable_all", true);
                   fmt.putAnnotation("enable_all", true);
                   fmt.setHidden(true);
                   fmt.setType(FORMAT_PHRASE);
                }
             }
```

Before astFactory.addASTChild(allPair, wid); is executed, the #a AST has the correct children; after is executed, the children are replaced with the i.e. Field widget AST.

The new field widget children I think should be siblings of the EXCEPT node... any idea why the above is not working as intended?

#60 - 10/09/2017 03:54 PM - Greg Shah

I suspect the direct manipulation of the #a node as the root node is the problem. Do we really need to manipulate that? The direct use of setNextSibling() and setFirstChild() can attach new nodes into the KW_ALL parent.

#61 - 10/09/2017 05:59 PM - Constantin Asofiei

A status what's left:

1. ENABLE ALL must not pull schema HELP for a field referenced by EXCEPT (VALEXP works, just HELP is problematic)
2. COLOR statement is implemented for GUI, what is left is to check the #3281-49 abbreviations

#62 - 10/10/2017 08:41 AM - Constantin Asofiei

I fixed last issues related to schema VALEXP/HELP and USE-DICT-EXPS; and also I've checked the color abbreviations, too.

Greg/Eric: please review 3281a 11181 ; runtime testing is in progress.

#63 - 10/10/2017 10:41 AM - Greg Shah

Code Review Task Branch 3281a Revision 11181

I think I see the beginnings of the work for "GUI and ChUI can run at the same time". :) It looks good.

My only question: in ui_statements.rules should removeQuotes() be used or should it be progressToJavaString() (in the case of a STRING node)?

#64 - 10/10/2017 10:43 AM - Constantin Asofiei

Greg Shah wrote:

> My only question: in ui_statements.rules should removeQuotes() be used or should it be progressToJavaString() (in the case of a STRING node)?

I chose removeQuotes() because 4GL allows both color display red/green and color display "red/green". Although progressToJavaString looks more complete.

#65 - 10/10/2017 02:22 PM - Constantin Asofiei

- Related to Bug #3347: HELP string support doesn't work for demo_widgets.p added

#66 - 10/10/2017 03:18 PM - Greg Shah

Code Review Task Branch 3281a Revision 11182

The change does not look correct. Won't frameNode be null if the widref.type == prog.expression?

#67 - 10/10/2017 03:30 PM - Constantin Asofiei

Greg Shah wrote:

> Code Review Task Branch 3281a Revision 11182
>
> The change does not look correct. Won't frameNode be null if the widref.type == prog.expression?

See 11183... the code in 11182 was completely off with what I had in mind to do.

#68 - 10/10/2017 03:51 PM - Constantin Asofiei

There is still one regression at least related to ENABLE ALL... I've fixed one issue (SymbolResolver.addFrameFields was not collecting unique frame fields, it was collecting all references. And there is one more: for some reason, after ENABLE ALL has the fields with schema validation/help pulled as children for ENABLE, they are "leaked" to the previous statement - i.e. a DISPLAY will include it, even if it was not targeted by that; don't know yet why is happening.

#69 - 10/10/2017 04:23 PM - Constantin Asofiei

Greg, please review 3281a 11184 - it solves the last found regression.

#70 - 10/10/2017 04:31 PM - Greg Shah

Code Review Task Branch 3281a Revision 11184

It looks good.

Were you going to change ui_statements.rules to use progressToJavaString() instead of removeQuotes()? I think it is safer in case someone encodes embedded quotes, escape chars etc..

#71 - 10/10/2017 04:43 PM - Constantin Asofiei

Greg Shah wrote:

> Were you going to change ui_statements.rules to use progressToJavaString() instead of removeQuotes()? I think it is safer in case someone encodes embedded quotes, escape chars etc..

Well, I had to use both, removeQuotes() to get rid of the quotes and progressToJavaString() to process the string (which is ensured to be explicitly quoted before the call). See 3281a rev 11185.

#72 - 10/10/2017 05:27 PM - Greg Shah

Code Review Task Branch 3281a Revision 11185

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

#73 - 10/11/2017 09:22 AM - Constantin Asofiei

Stanislav, a quick question please: is there a way to enable a browse column at runtime, via something similar to SENSITIVE attribute? I want to avoid using the ENABLE section of DEFINE BROWSE.

#74 - 10/11/2017 09:53 AM - Constantin Asofiei

There is still something else wrong in attachSchemaValidation and how is used, especially in BROWSE case:

1. ASTPair anchor argument is used to append a child via astFactory.addASTChild(anchor, valParent);; but the addASTChild, on each call, will just set the anchor.child reference to whatever the second argument is - so calling addASTChild more than once will override the previous set anchor.child. We need to have a parent FORMAT_PHRASE AST and call addChild on this AST directly...
2. when called via column_spec or column_ref (for BROWSE), this uses current_AST as anchor, which is i.e. the DISPLAY in DEFINE BROWSE DISPLAY - this is incorrect, the anchor must always be a format phrase AST (and if doesn't exist, create one)
3. I haven't check deeper, but attachAllAssignSchemaValidation creates only one FORMAT_PHRASE node, which is used as an anchor for ALL FIELDS (and their HELP/VALEXP)! Shouldn't there be one FORMAT_PHRASE AST for each field which has its valexp/help pulled from the schema?

I'm trying to fix the above now.

#75 - 10/11/2017 09:56 AM - Greg Shah

> Shouldn't there be one FORMAT_PHRASE AST for each field which has its valexp/help pulled from the schema?

Yes, definitely.

#76 - 10/11/2017 10:04 AM - Stanislav Lomany

- File BrowseColumnWidget.diff added

- File BrowseWidget.diff added

> is there a way to enable a browse column at runtime,

Yes.
1. Set READ-ONLY attribute to FALSE for the browse.
2. Set READ-ONLY attribute to FALSE for the browse column you want to enable.
3. You need to apply some changes from 3275c. I tried to made the patch. It is attached.

#77 - 10/11/2017 10:07 AM - Constantin Asofiei

Stanislav Lomany wrote:

> > is there a way to enable a browse column at runtime,
>
> Yes.
> 1. Set READ-ONLY attribute to FALSE for the browse.
> 2. Set READ-ONLY attribute to FALSE for the browse column you want to enable.
> 3. You need to apply some changes from 3275c. I tried to made the patch. It is attached.

Do you have a simple 4GL snippet? Because trying to set READ-ONLY for a browse column does not work, if the column is not referenced in the browse's ENABLE section.

#78 - 10/11/2017 10:13 AM - Stanislav Lomany

- File br-dyn-val-expr.p added

> Do you have a simple 4GL snippet?

Attached.

#79 - 10/11/2017 10:23 AM - Constantin Asofiei

Stanislav Lomany wrote:

> > Do you have a simple 4GL snippet?
>
> Attached.

So a column for a static browse can't change READ-ONLY if not part of ENABLE, correct? And only dynamic browse can work with READ-ONLY.

#80 - 10/11/2017 10:35 AM - Stanislav Lomany

You cannot do that for a static browse in 4GL ("cannot change READ-ONLY ... if ENABLE phrase is not specified"). But you CAN do that in P2J at this point.

```
def temp-table tt field f1 as integer
                 field f2 as character.

def var i as integer.
repeat i = 1 to 20:
   create tt. tt.f1 = i. tt.f2 = "Test " + string(i).
end.

DEFINE QUERY q FOR tt SCROLLING.
OPEN QUERY q FOR EACH tt.

DEF BROWSE br QUERY q
DISPLAY
     tt.f1
     tt.f2
   WITH size-chars 65 by 10 TITLE "Static browse" no-auto-validate.

DEF FRAME fr-orig br SKIP(1)
WITH TITLE "Frame" SIZE 70 BY 15 NO-LABELS.
```

```
ENABLE ALL WITH FRAME fr-orig.

browse br:read-only = false.
browse br:first-column:read-only = false.

WAIT-FOR WINDOW-CLOSE OF DEFAULT-WINDOW.
```

#81 - 10/12/2017 05:06 PM - Constantin Asofiei

Greg, I think I have a fix for the BROWSE issues - please see 3281a rev 11188. rev 11189 fixes the SymbolResolver.convertSeparator for Windows OS, I'll release it with this branch if you don't want to include it in another one.

Runtime testing is inconclusive at this time, will see tomorrow morning how it works.

#82 - 10/12/2017 05:20 PM - Greg Shah

Code Review Task Branch 3281a Revision 11189

I'm fine with the changes. They are much easier to understand now.

#83 - 10/12/2017 05:21 PM - Greg Shah

> rev 11189 fixes the SymbolResolver.convertSeparator for Windows OS, I'll release it with this branch if you don't want to include it in another one.

Yes, release it with this branch.

#84 - 10/12/2017 05:21 PM - Greg Shah

If it passes testing you can merge to trunk.

#85 - 10/18/2017 07:45 AM - Constantin Asofiei

- % Done changed from 70 to 100

Greg Shah wrote:

> If it passes testing you can merge to trunk.

After some more fixes, 3281a was merged to trunk rev 11178 and archived.

#86 - 10/18/2017 08:28 AM - Greg Shah

- Status changed from WIP to Closed

#87 - 10/19/2017 06:34 AM - Constantin Asofiei

Somehow I forgot to update the branch before merging - trunk 11179 completes this branch.

#88 - 03/04/2018 09:01 AM - Greg Shah

The frame option COLOR is marked as stubs for runtime. I believe it is fully supported now, right?

#89 - 03/04/2018 09:29 AM - Constantin Asofiei

Greg Shah wrote:

> The frame option COLOR is marked as stubs for runtime. I believe it is fully supported now, right?

Actually I think I missed the runtime for this and other widgets except FILL-IN/TEXT/LABEL, but it should be easy to implement.

#90 - 03/05/2018 05:18 PM - Greg Shah

How much effort?

#91 - 03/06/2018 02:09 AM - Constantin Asofiei

Around 4 hours to fix.

#92 - 03/06/2018 08:38 AM - Greg Shah

It should probably be cleared now.

#93 - 05/02/2018 02:23 PM - Constantin Asofiei

Anyone has any idea why this code was added to FillInGuiImpl and EditorGuiImpl.mouseClicked?

```
      if (!(activePopupMenu instanceof EditorPopupGuiImpl) || !activePopupMenu.isDisplayed())
      {
         return;
      }
```

This prevents caret change via mouse click.

#94 - 05/02/2018 02:49 PM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Anyone has any idea why this code was added to FillInGuiImpl and EditorGuiImpl.mouseClicked?
> [...]
>
> This prevents caret change via mouse click.

It seems that in FillinGuiImpl this code has been accidentally added from EditorGuiImpl when I fixed text selection in fill-in.

#95 - 05/02/2018 03:40 PM - Sergey Ivanovskiy

Sergey Ivanovskiy wrote:

> Constantin Asofiei wrote:
>
> > Anyone has any idea why this code was added to FillInGuiImpl and EditorGuiImpl.mouseClicked?
> > [...]
> >
> > This prevents caret change via mouse click.
>
> It seems that in FillinGuiImpl this code has been accidentally added from EditorGuiImpl when I fixed text selection in fill-in.

Sorry, it seems this code has been added recently by rev 11251. Hynek should know why it works here.

#96 - 05/03/2018 11:33 AM - Hynek Cihlar

Sergey Ivanovskiy wrote:

> Sorry, it seems this code has been added recently by rev 11251. Hynek should know why it works here.

Yes, this mine. I will fix it.

#97 - 05/03/2018 03:34 PM - Hynek Cihlar

Hynek Cihlar wrote:

> Sergey Ivanovskiy wrote:
>
> > Sorry, it seems this code has been added recently by rev 11251. Hynek should know why it works here.
>
> Yes, this mine. I will fix it.

Fixed in 3538a revision 11256.

#98 - 05/03/2018 05:42 PM - Constantin Asofiei

The remaining issues were fixed as part of #3528, in 3507a rev 11291

#99 - 04/22/2019 10:21 AM - Greg Shah

The gap marking for the color phrase seems wrong (from rules/gaps/user_interface.rules):

```
<rule>opts.put(prog.kw_color,    rw.cvt_lvl_full    | rw.rt_lvl_stub)</rule> <!-- color option is stubbed on r
untime level -->
```

Shouldn't this be rt_lvl_full?

#100 - 04/22/2019 01:07 PM - Constantin Asofiei

Greg Shah wrote:

> The gap marking for the color phrase seems wrong (from rules/gaps/user_interface.rules):
>
> [...]
>
> Shouldn't this be rt_lvl_full?

Yes, it should be rt_lvl_full - I can't find anything in the code which would suggest more work is needed.

Files

color_phrase.png	10.8 KB	10/06/2017	Constantin Asofiei
BrowseColumnWidget.diff	784 Bytes	10/11/2017	Stanislav Lomany
BrowseWidget.diff	3.44 KB	10/11/2017	Stanislav Lomany
br-dyn-val-expr.p	1.85 KB	10/11/2017	Stanislav Lomany