Conversion Tools - Feature #1521

Feature # 1511 (New): Reporting v3

implement gap analysis rules or backing database/marker approach

09/10/2012 12:25 PM - Greg Shah

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	60.00 hours
Target version:	Reporting 3.0		
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Related to Conversion Tools - Feature #1515: implement gap analysis views in ...	**New**	**09/10/2012**

History

#1 - 09/26/2012 08:05 AM - Greg Shah

- Target version set to Milestone 2

- Estimated time set to 60.00

#2 - 10/31/2012 01:25 PM - Greg Shah

- Assignee set to Greg Shah

#3 - 11/21/2012 09:05 AM - Greg Shah

- Target version changed from Milestone 2 to 25

#4 - 05/03/2016 01:19 PM - Greg Shah

A first pass at this support is now implemented (it went in to the trunk with revision 11021). What is left to do:

1. Expand all the marking (see rules/gaps/expressions.rules and rules/gaps/gap_analysis_marking.xml) to include all language features. The current approach only supports built-in functions and methods/attributes.

2. As new marking support is implemented, update the rules/reports/profile.rpt to include a line similar to this one supportLvlExpr="execLib("read_support_level", this)" for any report that needs to add the support level columns. The expression should be changed to properly map the node with the annotation (marker) to the called read_support_level function's parameter.

#5 - 11/16/2016 01:34 PM - Greg Shah

- Target version changed from 25 to Reporting 3.0

#6 - 01/17/2017 11:24 AM - Greg Shah

- Related to Feature #1515: implement gap analysis views in all reports added

#7 - 01/17/2017 12:05 PM - Greg Shah

- Status changed from New to WIP

- Assignee changed from Greg Shah to Eugenie Lyzenko

- Start date deleted (09/10/2012)

- % Done changed from 0 to 20

Eugenie: You and I will be working on this together. It is time to complete this.

Todos

1. ~~Make sure that the support in expressions.rules is up to date with any changes that have occurred since trunk rev 11021.~~
2. Add text comment support (see below for details).
3. Add exception rules support (see below for details).
4. Add support (marking rules) for:
    - ~~frame options and frame phrases~~
    - ~~browse options~~
    - ~~widget options and format phrases~~
    - ~~language statements~~ and associated options
    - control flow constructs
    - block options
    - ~~i/o options~~
    - ~~variable types~~
    - ~~field types~~
    - define variable options
    - parameter modes and qualifiers
    - function calls to user defined functions
    - ~~all global variables (these are often called "built-in functions" in the ABL docs but take no parameters and can be used without parenthesis, so they "look like" a global variable)~~
    - confirm that we already handle all the built-in functions that have non-standard syntax (for examples see record_funcs, if_func or postfix_funcs in the parser)
    - widget types
    - event types
    - all system handles
    - ~~metadata usage~~
    - ~~embedded sql~~
    - differentiation of WHERE clause usage of expression elements
    - COLON and chaining
    - DB_REF_NON_STATIC, DB_SYMBOL, FILENAME
    - all lvalue types and qualifiers
    - schema elements
5. Add support for generating textile output (see below).

Comment Support

When something is partial or restricted (or maybe any other support level) we want to be able to encode some comment text to explain the specific limitations or problems to be aware of. This would be displayed in the reports (either as an additional column or as a link that can be displayed on demand). We probably need separate comments for conv and runtime.

Exception Rules

Often, it is the case that we have almost 100% support for a feature but because of some rare case, we must mark it as "partial". We want to be able to write rules that are specific to a given value (not all built-in functions, but just CAN-FIND) and that rule can calculate the support level. The idea is that sometimes it would be much better to be able detect the exact unsupported cases instead of just coding something as "partial".

Textile Support

The idea is that we want to generate textile outout from these rules which can be pasted into the official documentation so that we can auto-generate support tables in our docs. This way we just have to maintain the rules in the rules/gaps/ directory and it is easy to keep our main docs up to date. This will require us to encode additional data in these rules so that we can generate everything properly. It will also require a special rule-set that gets used with these regular gap analysis rule-sets, which is used just for generating the textile output. It would not be used for real reporting.

#8 - 01/18/2017 07:16 PM - Eugenie Lyzenko

Let me clarify the point number 1 in in #1521-7.

The expression.rules file is a list of 4GL language elements(literals, operators, built-in functions, global variables, attributes and methods, misc expressions) we currently support. Each one has declared support status for conversion and runtime. At the time this rule file was committed not all of them had *_lvl_full status. But now it is possible the some status could changed but this possibly is not reflected in expression.rules file.

So the task is to check all elements in expression.rules that with not *_lvl_full status and verify if the current support has been upgraded and the file expression.rules is need to be changed. To do this it is required to scan full P2J project(or changes since 11021) for possible new changes for support level of the language elements that are not in expression.rules.

Have I missed for something? Or wrong understanding somewhere?

#9 - 01/18/2017 07:23 PM - Greg Shah

I think you have the right idea. I don't need you to review every line of code change. Instead, I just want you to read through the bzr log -v -r11021 to find anything that is changed but not yet coded properly.

#10 - 01/19/2017 01:57 PM - Eugenie Lyzenko

I have found the RESIZABLE attribute has runtime support. So at least one change is required for expression.rules file.

So going to create 1521a branch in 15 min. if there are no objections.

#11 - 01/19/2017 02:44 PM - Eugenie Lyzenko

Created task branch 1521a from trunk revision 11136.

#12 - 01/19/2017 09:08 PM - Eugenie Lyzenko

I have scanned the files from bzr log -v -r11021. In addition I have looked into p2j/ui/* directory(with subdirs) and here it what differences I have found:
1. CURRENT-ITERATION attribute. We have the getter implemented and stub for setter, the expressions.rules record:

```
<rule>attrs.put(prog.kw_cur_iter, rw.cvt_lvl_full        | rw.rt_lvl_stub)</rule>
```

2. MOVABLE attribute has runtime support, the expressions.rules record:

```
<rule>attrs.put(prog.kw_movable , rw.cvt_lvl_full        | rw.rt_lvl_stub)</rule>
```

3. RESIZABLE attribute has runtime support(properly setting the config value), the expressions.rules record:

```
<rule>attrs.put(prog.kw_resizabl, rw.cvt_lvl_full        | rw.rt_lvl_none)</rule>
```

My suggestion for changing the expressions.rules:

```
<rule>attrs.put(prog.kw_cur_iter, rw.cvt_lvl_full        | rw.rt_lvl_partial)</rule>
...
<rule>attrs.put(prog.kw_movable , rw.cvt_lvl_full        | rw.rt_lvl_full)</rule>
...
<rule>attrs.put(prog.kw_resizabl, rw.cvt_lvl_full        | rw.rt_lvl_full)</rule>
```

Let me know if this is OK or not.

#13 - 01/20/2017 08:56 AM - Eugenie Lyzenko

Rebased task branch 1251a from P2J trunk revision 11137. Still no changes since branch creation.

#14 - 01/20/2017 09:21 AM - Greg Shah

<rule>attrs.put(prog.kw_cur_iter, rw.cvt_lvl_full | rw.rt_lvl_partial)</rule> seems correct. Please put a comment at the end of that line about the missing setter.

For MOVABLE and RESIZABLE, I think rw.rt_lvl_partial is more correct. I don't think we handle all use cases yet. Please check the changes again. My understanding is that we may only have a small amount of support (maybe only for 1 or 2 widgets, even then it may be partial). Both of these are used for "direct manipulation", which include drag and drop. This includes use cases where the user can actually pick up the widget and move it or select the widget (when SELECTABLE = true) and change its size. This is used in the appbuilder (which is written in the 4GL itself) to allow graphical editing of 4GL programs. Try to identify the exact case that we do support and add comments to expressions.rules on those lines.

#15 - 01/20/2017 10:12 AM - Eugenie Lyzenko

Greg Shah wrote:

> <rule>attrs.put(prog.kw_cur_iter, rw.cvt_lvl_full | rw.rt_lvl_partial)</rule> seems correct. Please put a comment at the end of that line about the missing setter.

OK.

> For MOVABLE and RESIZABLE, I think rw.rt_lvl_partial is more correct. I don't think we handle all use cases yet. Please check the changes again. My understanding is that we may only have a small amount of support (maybe only for 1 or 2 widgets, even then it may be partial). Both of these are used for "direct manipulation", which include drag and drop. This includes use cases where the user can actually pick up the widget and move it or select the widget (when SELECTABLE = true) and change its size. This is used in the appbuilder (which is written in the 4GL itself) to allow graphical editing of 4GL programs. Try to identify the exact case that we do support and add comments to expressions.rules on those lines.

Yes, RESIZABLE has very limited support, actually only for BrowseColumnWidget which is as far as I know not a real 4GL widget. For all other widget classes it works like a stub(but without generation the "Unsupported..." error message and exception)

The MOVABLE support is limited to the FRAME widget and only for the server side(by setting proper value for config.movable boolean). The client side do nothing with this flag.

So you are right, these attributes need to be declared as having rw.rt_lvl_partial for runtime support status.

#16 - 01/20/2017 11:03 AM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11138.

The expressions.rules updated to change the supported level flag to rw.rt_lvl_partial for all 3 attributes noted before. The respective comments have also been added to the rule file.

#17 - 01/20/2017 12:01 PM - Greg Shah

Code Review Task Branch 1521a Revision 11138

The changes are good.

#18 - 01/20/2017 04:39 PM - Eugenie Lyzenko

Clarification for the implementation point number 5 in note #1521-7.

As far as I understand we need to use the data provided by expression.rules file to generate the textile tables like this:

Attribute	Conversion Support Level	Runtime Support Level	Notes

And this is not directly related to the currently converted 4GL application itself so the output must be placed somewhere within working P2J tree. Correct?

Currently the report generation is performed on the one of the conversion process step by calling ReportWorker$Library.generateAllReports() method from one of the report related rule, for example: reports/consolidated_reports.xml. Correct?

So we need to create a rule that will become a part of the conversion process. The rule will take the data from expression.rules, collect them and call something like: ReportWorker.generateTextileReports() to produce output that can be inserted into our page like https://proj.goldencode.com/projects/p2j/wiki/Supported_Features. Correct?

#19 - 01/20/2017 04:54 PM - Greg Shah

> As far as I understand we need to use the data provided by expression.rules file to generate the textile tables like this:

I edited the table in note 18 to show some changes. The idea of the table is correct.

> And this is not directly related to the currently converted 4GL application itself

Yes.

> so the output must be placed somewhere within working P2J tree. Correct?

Not necessarily. I think it can just be placed in the current directory.

> Currently the report generation is performed on the one of the conversion process step by calling ReportWorker$Library.generateAllReports() method from one of the report related rule, for example: reports/consolidated_reports.xml.

Yes, but that is not needed here.

> So we need to create a rule that will become a part of the conversion process.

No, I don't think it needs to run each time conversion runs. My idea was to only run it when the gap analysis rules (like expressions.rules) change.

> The rule will take the data from expression.rules, collect them and call something like: ReportWorker.generateTextileReports()

Almost. We don't need to use any of the ReportWorker functionality. We can just create a rule-set similar to gap_analysis_marking.xml which includes the other gap analysis rules. Call it rules/gaps/generate_textile_tables.xml. In that rule-set, it would have TRPL code that can iterate through the maps of data (like the attrmeth hash-map) and then use the file output functions in CommonAstSupport (see rules/convert/brew.xml for examples and search on deleteFile, openStream, fprintf and closeStream. You can directly write the files yourself, from the TRPL rules. That is why there is no need for the ReportWriter.

Whenever the gap analysis rules change, we can run this rule-set using the PatternEngine. Each table can be output to a file with its own filename. Then we just manually cut and paste the textile output into the documentation pages in Redmine.

#20 - 01/20/2017 04:56 PM - Greg Shah

One tricky thing is that the PatternEngine will require us to pass in an AST when we run it. But all these rules don't need to walk a real tree, as you noted. So any tree will work, we just need to process everything in the init or post rules.

#21 - 01/20/2017 05:42 PM - Eugenie Lyzenko

Greg Shah wrote:

> One tricky thing is that the PatternEngine will require us to pass in an AST when we run it. But all these rules don't need to walk a real tree, as you noted. So any tree will work, we just need to process everything in the init or post rules.

OK.

So the textile table generation should work even if we have no 4GL code to convert? We need to pass some artificial AST just for the PatternEngine to be able to start and execute, right?

I would suggest to make separate *.sh script to generate textiles by calling PatternEngine. So the new rules/gaps/generate_textile_tables.xml rule file will be processing only by this script but not involving as a stage of the 4GL code conversion cycle.

#22 - 01/20/2017 06:06 PM - Greg Shah

Yes. That makes sense.

#23 - 01/21/2017 07:27 AM - Greg Shah

In preference to item 5 of note 7, please work on these parts of item 4:

- frame options and frame phrases
- browse options
- widget options and format phrases

Carefully go through the progress.g and pull out the list of the above options (including the undocumented ones). Then research each one on both the conversion and runtime. For any limitations that you find, please leave a comment on the same line.

Put these into a new rule-set rules/gaps/user_interface.rules and you can link it into gap_analysis_marking.xml similarly to how expressions.rules is used. You'll have to create the maps in gap_analysis_marking.xml in the same way as well.

Once these new rules are implemented, please update rules/reports/profile.rpt to enable the support level columns in the UI options reports by adding the supportLvlExpr as documented in note 4.

#24 - 01/23/2017 04:13 AM - Eugenie Lyzenko

Greg Shah wrote:

> In preference to item 5 of note 7, please work on these parts of item 4:
>
> - frame options and frame phrases
> - browse options

- widget options and format phrases

Carefully go through the progress.g and pull out the list of the above options (including the undocumented ones). Then research each one on both the conversion and runtime. For any limitations that you find, please leave a comment on the same line.

Put these into a new rule-set rules/gaps/user_interface.rules and you can link it into gap_analysis_marking.xml similarly to how expressions.rules is used. You'll have to create the maps in gap_analysis_marking.xml in the same way as well.

Once these new rules are implemented, please update rules/reports/profile.rpt to enable the support level columns in the UI options reports by adding the supportLvlExpr as documented in note 4.

OK.

#25 - 01/23/2017 03:24 PM - Eugenie Lyzenko

The question:

If other phrases are the parts of the frame phrase or frame option - should it be also added into user_interface.rules file as frame option?

For example: Frame has AT clause with options, do we need all options of the AT to be added as options of a frame?

#26 - 01/23/2017 05:30 PM - Greg Shah

I guess the answer depends on whether there are child nodes that represent unsupported configurations. For now, just implement the matching at the same level as the reports search (e.g. evalLib("frame_options", this)). That way the matching reports will be correct.

Please see item 3 in note 7 (exception support). That idea is one way to handle some exceptions.

At some point we will want to ensure that all the child nodes are marked so that we can calculate the percentage of code that is supported. But for now, just marking it at the top level is probably OK.

#27 - 01/23/2017 09:30 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11139. Rebased with trunk 11138, new version is 11140.

Created new file user_interface.rules. Added first steps of the frame and browse options and using them in gap_analysis_marking.xml. Continue working with remaining options.

#28 - 01/24/2017 01:18 PM - Greg Shah

I think some updates to the supported attributes are still missing.

Both prog.kw_col_mov and prog.kw_col_res are now fully supported (rw.rt_lvl_full). The changes came from #3038 in rev 10999 and the original

coding was wrong. Please look at the #3038 attrs/methods and see what needs to be updated.

#29 - 01/24/2017 01:40 PM - Eugenie Lyzenko

Greg Shah wrote:

> I think some updates to the supported attributes are still missing.
>
> Both prog.kw_col_mov and prog.kw_col_res are now fully supported (rw.rt_lvl_full). The changes came from #3038 in rev 10999 and the original coding was wrong. Please look at the #3038 attrs/methods and see what needs to be updated.

OK. I'll check.

BTW. I was able to access the task history entries **without** logging the site in(although only in reading mode). I have clicked on page link inside mail body and just got into https://proj.goldencode.com/issues/3038. Previously the site requested to password based login to go inside task record. Is it normal? Our internals are now visible for everyone?

#30 - 01/24/2017 02:13 PM - Greg Shah

> Is it normal?

Yes, for the FWD project (and TRPL and Harness).

> Our internals are now visible for everyone?

Yes. It is part of our open source preparations. It is also why we must be very careful to exclude private or customer data from those projects.

#31 - 01/24/2017 02:19 PM - Greg Shah

Method kw_mov_col is now fully supported too.

#32 - 01/24/2017 02:26 PM - Eugenie Lyzenko

Greg Shah wrote:

> Method kw_mov_col is now fully supported too.

OK. I looked inside the task and one point left not clear: CREATE-ON-ADD (kw_creat_oa).

Is it method, attribute or widget creation option? Only browse related? GUI only or both? The 4GL documentation does not tell anything.

The other words to verify and add:
browse gui FIT-LAST-COLUMN (kw_fit_lcol)
browse gui COLUMN-MOVABLE (kw_col_mov)
browse gui COLUMN-RESIZABLE (kw_col_res)
browse MIN-HEIGHT-CHARS (kw_min_h_c)
browse gui MOVE-COLUMN (kw_mov_col)

#33 - 01/24/2017 02:39 PM - Greg Shah

kw_creat_oa is an attribute. We have some support. You'll have to check the details.

#34 - 01/24/2017 03:48 PM - Eugenie Lyzenko

The attribute MIN-HEIGHT-CHARS is supported but only by setting config.minHeightChars value. Actually this value is not used in P2J. Does it mean the current runtime value rw.rt_lvl_partial is correct? Or rw.rt_lvl_stub fits better?

#35 - 01/24/2017 04:04 PM - Eugenie Lyzenko

The attribute CREATE-ON-ADD has limited support even less than MIN-HEIGHT-CHARS. It is implemented on the server side by properly set the config value via setter. The getter is also implemented on the server side. But both getter and setter are never calling.

What is the better value to describe this? stub or partial?

#36 - 01/24/2017 05:11 PM - Greg Shah

Both MIN-HEIGHT-CHARS and CREATE-ON-ADD can be marked as rw.rt_lvl_stub.

#37 - 01/24/2017 05:56 PM - Eugenie Lyzenko

Greg Shah wrote:

> Both MIN-HEIGHT-CHARS and CREATE-ON-ADD can be marked as rw.rt_lvl_stub.

Task branch 1521a for review updated to revision 11141.

Refreshing status for attributes FIT-LAST-COLUMN, COLUMN-MOVABLE, COLUMN-RESIZABLE, CREATE-ON-ADD, MIN-HEIGHT-CHARS and method MOVE-COLUMN.

The option NO-SCROLLBAR-VERTICAL will be handled later in other file containing browse related options.

#38 - 01/24/2017 09:45 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11142.

This is update for browse widget options status. Not all noted in progress.g have full conversion and runtime support. I need to carefully check everything to have proper data. My approach is:
1. Check rules for keyword to find out if keyword id involved into conversion.
2. Once the conversion support detected and runtime counterpart is found look for the source files to find out if the server and client sides operate with the converted constructs.
3. Depending on results have been found change the support level to none, stub, partial or full.

Currently browse widget handled up to kw_multiple keyword.

#39 - 01/25/2017 07:38 AM - Greg Shah

Ask Stanislav if you have any questions about the status of a browse option.

#40 - 01/25/2017 09:13 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11143.

Completed handling of the browse widget options/attributes. Continue with frame options.

#41 - 01/26/2017 11:16 AM - Eugenie Lyzenko

Found the attributes PAGE-BOTTOM and PAGE-TOP are both fully supported for conversion and runtime.

Do I need to change the expressions.rules file that currently defines rw.(cvt|rt)_lvl_none for these attributes?

#42 - 01/26/2017 11:23 AM - Greg Shah

Are you talking about the attributes or the options? Although the 4GL generally implements both with the same backing code, we cannot ever be sure.

And regardless of that, we haven't yet added kw_page_t and kw_page_b to methods_attributes.rules. Although it would be easy to enable them because the backing functionality is there, I don't think we can mark these as supported.

However, it is worth putting a comment into expressions.rules on those lines: "the core runtime functionality is fully implemented for the corresponding frame option, but the attribute support at conversion and runtime has not yet been implemented"

#43 - 01/26/2017 11:51 AM - Eugenie Lyzenko

Greg Shah wrote:

> Are you talking about the attributes or the options? Although the 4GL generally implements both with the same backing code, we cannot ever be sure.

Both are the frame options, not attributes.

> And regardless of that, we haven't yet added kw_page_t and kw_page_b to methods_attributes.rules. Although it would be easy to enable them

> because the backing functionality is there, I don't think we can mark these as supported.
>
> However, it is worth putting a comment into expressions.rules on those lines: "the core runtime functionality is fully implemented for the corresponding frame option, but the attribute support at conversion and runtime has not yet been implemented"

OK.

#44 - 01/26/2017 12:03 PM - Greg Shah

> Both are the frame options, not attributes.

expressions.rules is where we document the attributes, not the options. Please be careful with your terminology so that we can avoid confusion. Since the 4GL often has overlap between options and attributes, the support levels can be different.

#45 - 01/26/2017 04:11 PM - Eugenie Lyzenko

Greg Shah wrote:

> > Both are the frame options, not attributes.
>
> expressions.rules is where we document the attributes, not the options. Please be careful with your terminology so that we can avoid confusion. Since the 4GL often has overlap between options and attributes, the support levels can be different.

OK. This terminology confusion cased by fact the options are frequently used to set up initial value of the widget attribute.

Task branch 1521a for review updated to revision 11144.

This is the refresh for support status gap rules for frame and browse widgets. Looks like I need one more pass through the 4GL documentation to find out whether I have missed for other options(I have some suspects I did it).

#46 - 01/26/2017 05:50 PM - Eugenie Lyzenko

I have found additional options to add to frame options status. It is related to FRAME phrase. But I need some clarifications:

1. FRAME option of the clause like FORM something WITH FRAME fr. In this case will the FRAME be the option of the FRAME phrase. Meaning do we need to put FRAME option to the frame options map?
2. What to do with VIEW-AS DIALOG-BOX option? Is it a valid option for frame widget in the current task?
3. What about IN WINDOW frame phrase option?

#47 - 01/26/2017 06:33 PM - Eugenie Lyzenko

After rescan 4GL documents found more options to add:

Frame phrase:

```
ACCUM
CANCEL-BUTTON
COLUMN
COLUMNS
CONTEXT-HELP-FILE
DEFAULT-BUTTON
DOWN
WIDGET-ID
FONT
FRAME ?
NO-AUTO-VALIDATE
RETAIN
ROW
SCROLL
STREAM
STREAM-HANDLE
VIEW-AS DIALOG-BOX ?
WIDTH
IN WINDOW ?
```

Browse options:

```
QUERY
SHARE-LOCK
EXCLUSIVE-LOCK
NO-LOCK
NO-WAIT
DISPLAY
EXCEPT
CONTEXT-HELP-ID
TOOLTIP
HELP
VALIDATE
AUTO-RETURN
DISABLE-AUTO-ZAP
DOWN
SIZE
SIZE-PIXELS
FGCOLOR
BGCOLOR
DCOLOR
PFCOLOR
LABEL-FONT
LABEL-DCOLOR
LABEL-FGCOLOR
LABEL-BGCOLOR
MULTIPLE
SINGLE
NO-ASSIGN
NO-ROW-MARKERS
```

```
NO-LABELS
NO-BOX
FONT
TITLE
ROW_HEIGHT-CHARS
ROW-HEIGHT-PIXELS
```

Starting to add.

#48 - 01/26/2017 09:22 PM - Greg Shah

> 1. FRAME option of the clause like FORM something WITH FRAME fr. In this case will the FRAME be the option of the FRAME phrase. Meaning do we need to put FRAME option to the frame options map?

Yes, because it is part of the frame phrase and we do support it. But it is just for naming the frame.

> 2. What to do with VIEW-AS DIALOG-BOX option? Is it a valid option for frame widget in the current task?

Yes, it is a valid option. And we also support it fully.

> 3. What about IN WINDOW frame phrase option?

Yes, this is a valid option and we support it fully.

#49 - 01/26/2017 09:24 PM - Greg Shah

> After rescan 4GL documents found more options to add:

I don't understand. These are all visible in the progress.g from frame_phrase and browse_options_phrase. Why were they missed on the first pass?

#50 - 01/27/2017 07:26 AM - Eugenie Lyzenko

Greg Shah wrote:

> > After rescan 4GL documents found more options to add:
>
> I don't understand. These are all visible in the progress.g from frame_phrase and browse_options_phrase. Why were they missed on the first pass?

Because in progress.g these option are noted indirectly. For example(frame_phrase):

```
...
         | (KW_ATTR | KW_NO_ATTR)
         | button_reference
         | KW_CENTER
...
```

for cancel and default buttons the frame_phrase refers to the other clause in the progress.g. And I missed because I saw clause inside clause instead of direct option. Sorry.

#51 - 01/27/2017 09:12 AM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11145.

Finished options support status adding into user_interface.rules for frame and browse widgets. Shifting to the other widgets to add.

#52 - 01/27/2017 10:21 AM - Eugenie Lyzenko

Task branch 1521a rebased with trunk 11139, new version is 11146.

#53 - 01/27/2017 11:01 AM - Eugenie Lyzenko

Task branch 1521a rebased with trunk 11140, new version is 11147.

#54 - 01/27/2017 12:14 PM - Eugenie Lyzenko

Thinking about widget options map implementation. And some questions I have got:
1. Widget options map should be separated from browse widget map?
2. Another word we will have 3 types of data, one for frame, one for browse and rest - for another widgets, right?
3. As an edge condition - may be we need to separate map for every widget?

I'm asking because the widgets has subset of common options and sometimes the same option has different support level for different widgets. I saw this for frame and browse widget. Do you think it is OK to collect all possible widgets option in a single map for multiple widgets?

#55 - 01/27/2017 12:23 PM - Greg Shah

> Widget options map should be separated from browse widget map?

Yes.

> Another word we will have 3 types of data, one for frame, one for browse and rest - for another widgets, right?

Yes.

> As an edge condition - may be we need to separate map for every widget?

Maybe later. For now please keep it as a single map for all non-browse, non-frame widgets.

#56 - 01/27/2017 09:59 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11148.

Added options from general format phrase into widget options map. Continue with particular widget specific options.

#57 - 01/30/2017 05:42 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11149.

Added options for widgets other than browse and frame. The following keywords were not included:

```
begin_reserved
block
bool_false
bool_true
class_def
colon
comma
constructor
datetime_literal
datetime_tz_literal
date_literal
db_ref_non_static
db_symbol?
dec_literal
destructor
dot
editing_block
end_reserved
expression
filename
function
inner_block
library_ref
lparens
minus
multiply
num_literal
plus
procedure
rparens
string
transaction_distinct
unknown_val
```

I have a doubt because the looks as language keywords, not UI specific. Let me know if I'm wrong.

Also I have prepared the profile.rpt file change to handle the widget options:
reports/profile.rpt, Line 1565:

```
...
   <report
      condition="evalLib(&amp;quot;ui_language_stmts&amp;quot;, this)"
      dumpType="parser"
      supportLvlExpr="execLib(&amp;quot;read_support_level&amp;quot;, this)"
      prefix="ui_language_stmts"
      title="UI Language Statements for frames browses and other widgets"
      categories="User Interface"/>
...
```

Again let me know if this change is wrong(it is yet not committed into branch).

The current support level for widgets option is full for both conversion and runtime. And it will take a time to carefully test them all(>200 lines) if we have some missing in support. All options was taken from progress.g, respective clauses so I guess at least we have full conversion support for them.

Suspending work here for issue 3228 icon drawing fixing.

#58 - 01/31/2017 09:37 AM - Greg Shah

I think there are some problems with how you are going about this task.

1. There still seems to be confusion about what is a widget option and what is not a widget option.

For reporting purposes, here is how we match on a widget option:

```
   <report
      condition="evalLib(&amp;quot;widget_options&amp;quot;, this)"
      dumpType="parser"
      prefix="widget_options"
      title="Widget Options"
      categories="User Interface"/>
```

The widget_options function is used to match. You can see this code in include/common-progress.rules:

```
      <function name="widget_options">
         <parameter name="target" type="com.goldencode.ast.Aast" />
         <variable  name="ttype"  type="java.lang.Integer" />
         <variable  name="ptype"  type="java.lang.Integer" />
         <return    name="match"  type="java.lang.Boolean" />

         <rule>ttype = target.type</rule>
         <rule>match = false</rule>

         <rule>target.parent != null
            <action>ptype = target.parent.type</action>
            <action on="false">ptype = -1</action>
         </rule>

         <!-- most common forms of format phrase -->
         <rule>ptype == prog.format_phrase
            <action>match = true</action>
         </rule>

         <!-- browse options  -->
         <rule>evalLib("browse_options", target)
            <action>match = true</action>
         </rule>

         <!-- browse column ref -->
         <rule>
            (ttype == prog.kw_help      or
             ttype == prog.kw_validate  or
             ttype == prog.kw_auto_ret  or
             ttype == prog.kw_dis_a_za)     and
            ptype == prog.column_ref
            <action>match = true</action>
         </rule>

         <!-- msg stmt field options -->
         <rule>
            target.relativePath("STATEMENT/KW_MSG/KW_VIEW_AS")  or
            target.relativePath("STATEMENT/KW_MSG/KW_UPDATE/KW_AUTO_RET")  or
            target.relativePath("STATEMENT/KW_MSG/KW_UPDATE/KW_FORMAT")  or
            target.relativePath("STATEMENT/KW_MSG/KW_SET/KW_AUTO_RET")  or
            target.relativePath("STATEMENT/KW_MSG/KW_SET/KW_FORMAT")
            <action>match = true</action>
         </rule>

         <!-- enable stmt constant form item options -->
         <rule>
            (ttype == prog.kw_format    or
             ttype == prog.kw_view_as   or
             ttype == prog.kw_at        or
             ttype == prog.kw_to        or
             ttype == prog.kw_bgcolor   or
             ttype == prog.kw_dcolor    or
             ttype == prog.kw_fgcolor   or
             ttype == prog.kw_pfcolor   or
             ttype == prog.kw_font)         and
```

```
            target.upPath("STATEMENT/KW_ENABLE")
            <action>match = true</action>
         </rule>

         <!-- UI options in a define variable or temp table define field -->
         <rule>
            (ptype == prog.define_variable  or
             ptype == prog.define_field)       and
            (ttype != prog.symbol       and
             ttype != prog.kw_as        and
             ttype != prog.kw_like      and
             ttype != prog.kw_extent    and
             ttype != prog.kw_global    and
             ttype != prog.kw_init      and
             ttype != prog.kw_decimals  and
             ttype != prog.kw_case_sen  and
             ttype != prog.kw_new       and
             ttype != prog.kw_not       and
             ttype != prog.kw_no_undo   and
             ttype != prog.kw_shared    and
             ttype != prog.kw_triggers)
            <action>match = true</action>
         </rule>

         <!-- define parameter stmt/assign trigger var def -->
         <rule>
            (ttype == prog.kw_format    or
             ttype == prog.kw_label     or
             ttype == prog.kw_col_lab)         and
            (target.upPath("STATEMENT/TRIGGER_PROCEDURE/KW_NEW")   or
             target.upPath("STATEMENT/TRIGGER_PROCEDURE/KW_OLD")   or
             ptype == prog.define_parameter)
            <action>match = true</action>
         </rule>

         <!-- options in define widget stmts -->
         <rule>
            (ptype == prog.define_button                       or
             ptype == prog.define_rectangle                    or
             ptype == prog.define_image                        or
             ptype == prog.kw_menu_itm                         or
             target.upPath("DEFINE_MENU/KW_MENU")              or
             target.upPath("DEFINE_MENU/KW_MENU/KW_SUB_MENU") or
             target.upPath("DEFINE_SUB_MENU/KW_SUB_MENU")     or
             target.upPath("DEFINE_SUB_MENU/KW_SUB_MENU/KW_SUB_MENU")) and
            (ttype != prog.symbol       and
             ttype != prog.kw_triggers  and
             ttype != prog.kw_rule      and
             ttype != prog.kw_skip      and
             ttype != prog.kw_menu_itm)
            <action>match = true</action>
         </rule>

      </function>
```

Since the browse options marking have been handled separately, those parts can be ignored. But the other parts must each be reviewed carefully. Let's look at the format_phrase case. You can see that we are matching on the condition that the prog.format_phrase is our parent node's token type. So we must ONLY look at the direct children (NOT grandchildren or great grandchildren...) of the prog.format_phrase.

The next step is to look in progress.g and find the format_phrase rule (around line 11867):

```
format_phrase [ String symName, Aast fld, boolean frameDef, boolean enableWidget ]
   :
      {
         astFactory.makeASTRoot(currentAST, #[FORMAT_PHRASE,"format phrase"]);

         String baseSymName = null;

         boolean matchSym = false;
         boolean schem    = false;

         // validate processing needs a special promotion of the table
```

```
         // because the expression can have otherwise ambiguous field
         // references
         if (fld           != null             &&
             fld.getType()  > BEGIN_FIELDTYPES &&
             fld.getType()  < END_FIELDTYPES)
         {
            // add a schema scope
            schem = true;
            sym.addSchemaScope(false);

            // remember the current field name (possibly) being validated
            validateFieldName = (String) fld.getAnnotation("schemaname");

            // promote the field's associated buffer
            sym.promoteTableName((String) fld.getAnnotation("bufname"), false, false);
         }
      }
      (
         options { generateAmbigWarnings = false; }
         :
           at_phrase
         | { frameDef && symName != null }?
           as_button_quirk[symName]
         | (as_clause[symName, false] | like_clause[symName, null, false])
         | (KW_ATTR | KW_NO_ATTR)
         | KW_AUTO_RET
         | KW_BLANK
         | (colon_constant | to_constant)
         | column_label
         | KW_DEBLANK
         | KW_DIS_A_ZA
         | format_string[false]
         | help_string
         | (label | KW_NO_LABEL)
         | KW_NO_TAB_S
         | KW_PASSWD_F
         | v:validate
         | ui_stuff
         | view_as_phrase[symName]
         | simple_when_clause

           // we have to accomodate the fact that we can have an inline
           // var def where the symbol is not defined in the calling code
           // (and passed in via the symName parm) but is part of the
           // at base field clause
         | {
              matchSym = (symName == null);
           }
           baseSymName = at_base_field_clause[matchSym]
           {
              if (matchSym && baseSymName != null)
              {
                 symName = baseSymName;
              }
           }
      )*
...
```

Here we see that the rule will create a FORMAT_PHRASE node (which is the same as prog.format_phrase in TRPL) and it will attach direct children using some inline keyword matches (e.g. KW_ATTR, KW_NO_ATTR, KW_AUTO_RET...) and using some rule references (e.g. at_phrase or help_string).

You must then look at the rule references. Here is at_phrase:

```
at_phrase
   :  KW_AT^
      (
           location_spec location_spec
           (
              options { generateAmbigWarnings = false; }
              :
              KW_COLON_AL | KW_LEFT_AL | KW_RIGHT_AL
```

```
            )?
          | numeric_literal
      )
   ;
```

Notice that the root node of the at_phrase rule will always be KW_AT. That is the node that will be the direct child of FORMAT_PHRASE. KW_AT will have some children BUT they can NEVER be direct children of FORMAT_PHRASE. For this reason, only KW_AT would be matched as a widget option.

You would continue going through all the possible matching in the widget_options function (not just prog.format_phrase but also the message statement, enable statement etc...) and each of the progress.g rules to find the exact list of what can match there.

I see from note 57, that you are going too deep. You should not ever be thinking that a destructor could be matched by the widget_options rule. In other words, I think you wasted time looking at things that don't matter to this specific case.

After you have a complete list, you should double check that it seems right by comparing it with the widget options that can be seen in the Progress docs. Again, you should look at the lists/syntax diagrams for format phrase, message statement, enable statement and other docs for the parts that we match in widget_options to see if the list is right.

2. There is confusion about the level of support for things. When something is not fully implemented, you must not mark it as rw.*_lvl_full. Here is one example (there are many others):

```
<rule>opts.put(prog.kw_auto_ret,  rw.cvt_lvl_full    | rw.rt_lvl_full)</rule> <!-- runtime support implemented
 for fill-in only -->
```

The Progress docs say that auto-return can also be used for a browse column. If we don't support it fully at runtime, shouldn't it be marked as partial? It is the core objective of the task to get these assessments exactly correct. Otherwise when we use this gap analysis report it is wrong and we don't find out about it until much later which could cost us time and money.

3. Just because something is in progress.g does NOT mean we have conversion support. It just means we can parse the code. Conversion support for widget options is in the downstream rule sets, mostly in frame_generator.xml. So we must look there to determine if the conversion is supported or not.

#59 - 02/07/2017 08:52 PM - Eugenie Lyzenko

I have prepared the new list of the widget's options:

```
KW_ACCEL
KW_AS
KW_AT
KW_ATTR
KW_AUTO_END
KW_AUTO_GO
KW_AUTO_RET
KW_BGCOLOR
KW_BLANK
KW_CASE_SEN
KW_COL
KW_COLON
KW_COLUMNS
KW_COL_BGC
KW_COL_CP
KW_COL_DC
KW_COL_FGC
KW_COL_FONT
KW_COL_LAB
KW_COL_PFC
KW_CTX_H_ID
KW_CVT_3D_C
KW_DCOLOR
KW_DEBLANK
KW_DECIMALS
KW_DEFAULT
KW_DISABLED
KW_DIS_A_ZA
KW_DROP_TAR
KW_EDGE_C
KW_EDGE_P
KW_FGCOLOR
KW_FILE
KW_FIL_NAME
KW_FLAT_BUT
KW_FONT
KW_FORMAT
KW_GRAPHIC
KW_GROUP_BX
KW_HELP
KW_IMAGE
KW_IMG_DOWN
KW_IMG_INS
KW_IMG_SZ
KW_IMG_SZ_C
KW_IMG_SZ_P
KW_IMG_UP
KW_LABEL
KW_LAB_BGC
KW_LAB_DC
KW_LAB_FGC
KW_LAB_FONT
KW_LAB_PFC
KW_LIKE
KW_MARG_EX
KW_MENU_BAR
KW_MENU_ITM
KW_MOU_PTR
KW_NOT
KW_NO_ATTR
KW_NO_CV_3D
KW_NO_FILL
KW_NO_FOCUS
KW_NO_LABEL
KW_NO_TAB_S
KW_NO_UNDO
KW_PASSWD_F
KW_PFCOLOR
KW_READ_ONL
KW_RET_SHAP
KW_ROUNDED
```

```
KW_ROW
KW_RULE
KW_SERIALZH
KW_SIZE
KW_SIZE_C
KW_SIZE_P
KW_SKIP
KW_SKIP
KW_ST_2_FIT
KW_SUB_MENU
KW_SUB_MENU
KW_SUB_M_H
KW_TITLE
KW_TO
KW_TOGGL_BX
KW_TOOLTIP
KW_TRANSPAR
KW_TRIGGERS
KW_TTCP
KW_VIEW_AS
KW_WHEN
KW_WID_ID
KW_XML_DTYP
KW_XML_NNAM
KW_XML_NTYP
```

Double checking for the support level.

#60 - 02/08/2017 11:18 AM - Eugenie Lyzenko

> The Progress docs say that auto-return can also be used for a browse column. If we don't support it fully at runtime, shouldn't it be marked as partial? It is the core objective of the task to get these assessments exactly correct. Otherwise when we use this gap analysis report it is wrong and we don't find out about it until much later which could cost us time and money.

Let me clarify one point.
1. It is supposed we have 3 separate maps for widgets level support for:
- frame widget
- browse widget

- all other widgets
2. Let's consider the last one map.
3. Take the option prog.kw_auto_ret. The option is valid for fill-in and browse(browse column). It has full support for fill-in widget and only stub level support for browse(browse column).
4. In the browse option map the support level will be: rw.cvt_lvl_full | rw.rt_lvl_stub.
5. But from the other widgets map(where only fill-in is considering) the support is: rw.cvt_lvl_full | rw.rt_lvl_full. Am I wrong?

Another word if the single option belongs to all frame, browse and some other widget - we need to define this option in all 3 maps with possible different support level, correct?

#61 - 02/08/2017 11:40 AM - Greg Shah

> But from the other widgets map(where only fill-in is considering) the support is: rw.cvt_lvl_full | rw.rt_lvl_full. Am I wrong?

Yes.

> Another word if the single option belongs to all frame, browse and some other widget - we need to define this option in all 3 maps with possible different support level, correct?

Yes. But we will need to exclude the browse from the widget_options function in include/common-progress.rules so that we don't improperly mark the browse option using the widget option map. In other words, the matching for the 3 maps must be mutually exclusive.

#62 - 02/08/2017 08:54 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11150.

Fix for UI options list status maps. This is the base for the further work here. Removed everything that was attribute related but not options. The attribute is not always available to be set up through option.

Continue working. Need to complete comments adding for all we have not full support and go to the next step in this task.

#63 - 02/09/2017 02:04 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11151.

More clean ups for user_interface.rules file. The options not having full support level are now with comments. Some options support level was reviewed, checked once again and fixed.

Continue working with using this rule file approach implementation.

#64 - 02/09/2017 05:44 PM - Eugenie Lyzenko

Please clarify me some points if my understanding is wrong.

1. The control file is reports/profile.rpt, current state:

```
...
   <report
      condition="evalLib(&amp;quot;frame_options&amp;quot;, this)"
      dumpType="parser"
      prefix="frame_options"
      title="Frame Options"
      categories="User Interface"/>
...
   <report
      condition="evalLib(&amp;quot;browse_options&amp;quot;, this)"
      dumpType="parser"
      prefix="browse_options"
      title="Browse Options"
      categories="User Interface"/>
...
   <report
      condition="evalLib(&amp;quot;widget_options&amp;quot;, this)"
      dumpType="parser"
      prefix="widget_options"
      title="Widget Options"
      categories="User Interface"/>
...
```

For support level to work is should be changed to:

```
...
   <report
      condition="evalLib(&amp;quot;frame_options&amp;quot;, this)"
      dumpType="parser"
      supportLvlExpr="execLib(&amp;quot;read_support_level&amp;quot;, this)"
      prefix="frame_options"
      title="Frame Options"
      categories="User Interface"/>
...
   <report
      condition="evalLib(&amp;quot;browse_options&amp;quot;, this)"
      dumpType="parser"
      supportLvlExpr="execLib(&amp;quot;read_support_level&amp;quot;, this)"
      prefix="browse_options"
      title="Browse Options"
      categories="User Interface"/>
...
   <report
      condition="evalLib(&amp;quot;widget_options&amp;quot;, this) and !evalLib(&amp;quot;browse_options&amp;quot;, this)"
      dumpType="parser"
      supportLvlExpr="execLib(&amp;quot;read_support_level&amp;quot;, this)"
      prefix="widget_options"
      title="Widget Options"
      categories="User Interface"/>
...
```

The last change is necessary to separate browse widget from all others. This change turn the report generation on(if rules in gap_analysis_marking.xml are implemented properly).

2. The next conversion with f2+m0+cb will generate the reports in rpt/code/frame_options, rpt/code/browse_options and rpt/code/widget_options respectively and I can see if the support level data are there to check the new rule set is working properly. Correct?

Am I missing some stages? No need to run full conversion and reports can be generated by separate command?

#65 - 02/09/2017 09:31 PM - Greg Shah

> I can see if the support level data are there to check the new rule set is working properly. Correct?

Yes.

> Am I missing some stages?

No.

> No need to run full conversion

Correct. You run the F2 (front end) only.

> and reports can be generated by separate command?

Correct.

Test it using the Hotel GUI project and with the most recent #3228 POC. ant rpt will parse and generate reports.

#66 - 02/10/2017 11:35 AM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11152.

This is the first working release of the widget options status report creation.

Reports generated. There are some issues to be clarified:
1. The frame_options report has item browse [KW_BROWSE] with unknown status for code DISPLAY ... WITH BROWSE...:

```
. . .
```

```
01852        IF AVAILABLE soldto THEN
01853           DISPLAY soldto.sold-id soldto.sold-name soldto.sold-city soldto.sold-state soldto.sold-zip WI
TH BROWSE Browser-Table
01854              NO-ERROR.
...
```

Looks like we have the browse widget that considering as option for frame widget. This means it must be moved to the browse widget options, right? We need to check why it is considering as frame instead of proper browse widget, correct?

2. The widget_options has item WID_MENU_ITM with unknown status for:

```
...
01640  ON CHOOSE OF MENU-ITEM m_Misc_Fields OR
01641     CHOOSE OF MENU-ITEM p_Misc_Fields
01642  DO:
01643    RUN Select_Misc_Fields.
01644  END.
...
```

The MENU-ITEM(WID_MENU_ITM) is considering as option for m_Misc_Fields. Is this correct? We need to either change widget option criteria or add rule:

```
...
         <rule>opts.put(prog.wid_menu_itm,       rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
...
```

3. The same as in point 2 but for AT reserved char:

```
...
00155        TODAY - ar-inv.inv-date @ li-days-old COLUMN-LABEL "Days Old" FORMAT "->>,>>>":U
...
```

The char @ is not a widget option, right? Need to fix the char is considering as widget option.

#67 - 02/10/2017 12:44 PM - Greg Shah

> Looks like we have the browse widget that considering as option for frame widget. This means it must be moved to the browse widget options, right? We need to check why it is considering as frame instead of proper browse widget, correct?

Correct.

> The MENU-ITEM is considering as option for m_Misc_Fields. Is this correct?

No, it is not an option. Please exclude it.

> The char @ is not a widget option, right?

No. Actual the AT (@) char IS a widget option.

#68 - 02/10/2017 05:49 PM - Eugenie Lyzenko

Almost all issues have fixed. One more point where I have some doubts.

The widget_options function considers KW_VALIDATE as valid widget option. But as far as I know this is not one the widget can take on creation/definition. This refers to validate() method, even not the attribute.

Correct me if I'm wrong validate is not correct widget option and should be removed from widget_options as match condition. Correct?

Example ./gui/viewers/cust.w:

```
...
 [KW_VALIDATE] @0:0
    [EXPRESSION] @0:0
      gt [GT] @1:22
         index [FUNC_INT] @1:1
            "AISX" [STRING] @1:7
            active [FIELD_CHAR] @1:14
         0 [NUM_LITERAL] @1:25
   null [EXPRESSION] @0:0
      "Must be (A)ctive, (I)nactive, (X) Inhouse, or (S)tatus" [STRING] @0:0
...
```

#69 - 02/10/2017 05:51 PM - Greg Shah

> This refers to validate() method, even not the attribute.

No, this is not the validate() method. It is really an option.

> Correct me if I'm wrong validate is not correct widget option and should be removed from widget_options as match condition. Correct?

No. It should be included as an option. We fully support it.

#70 - 02/10/2017 05:55 PM - Eugenie Lyzenko

Greg Shah wrote:

> > This refers to validate() method, even not the attribute.
>
> No, this is not the validate() method. It is really an option.
>
> > Correct me if I'm wrong validate is not correct widget option and should be removed from widget_options as match condition. Correct?
>
> No. It should be included as an option. We fully support it.

OK. Thanks for clarification.

#71 - 02/10/2017 07:31 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11153.

The changes:
1. MENU-ITEM(WID_MENU_ITM) removed from valid widget option.
2. BROWSE(KW_BROWSE) removed from frame option.
3. Adding @ and KW_VALIDATE as valid options into user_interface.rules
4. The common-progress.rules changed to separate browse column options in standalone function(the code is using more than once).
5. profile.rpt has changes to tune the report generation conditions. The browse column related option added as condition to browse options report generator. Also browse and browse column option excluded from general widget option report condition rule.

The report is now generating, tested on poc application. So need to make last check for options in user_interface.rules file. I just want to find a way to get more guarantee I've not missed something related to support level or option elements to add.

#72 - 02/11/2017 04:22 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11154.

Adding missed options after re-check with bigger project report generation.

#73 - 02/13/2017 10:20 AM - Greg Shah

I think we support the widget option kw_width at conversion (it is mapped to setWidthChars() in frame_generator.xml but it is marked as rw.cvt_lvl_none. Did I get that wrong?

#74 - 02/13/2017 10:43 AM - Eugenie Lyzenko

Greg Shah wrote:

> I think we support the widget option kw_width at conversion (it is mapped to setWidthChars() in frame_generator.xml but it is marked as rw.cvt_lvl_none. Did I get that wrong?

You are right. I considered width-(chars|pixels) - we have no conversion support for them(supported as attributes). My mistake. width option is supported at conversion.

Fixed. Task branch 1521a for review updated to revision 11155.

#75 - 02/13/2017 10:52 AM - Greg Shah

The result is good.

Next, please work on:

- all global variables (these are often called "built-in functions" in the ABL docs but take no parameters and can be used without parenthesis, so

they "look like" a global variable)
- confirm that we already handle all the built-in functions that have non-standard syntax (for examples see record_funcs, if_func or postfix_funcs in the parser)

#76 - 02/13/2017 11:44 AM - Eugenie Lyzenko

Greg Shah wrote:

> Next, please work on:
>
> - all global variables (these are often called "built-in functions" in the ABL docs but take no parameters and can be used without parenthesis, so they "look like" a global variable)
> - confirm that we already handle all the built-in functions that have non-standard syntax (for examples see record_funcs, if_func or postfix_funcs in the parser)

OK. Just to clarify the task.

The mapping for both already defined in expressions.rules file, all adding/change for mapping must be done in this file, correct?

In addition I need to modify gap_analysis_marking.xml to add global variables handling, built-in function processing already there, correct?

#77 - 02/13/2017 12:00 PM - Greg Shah

> The mapping for both already defined in expressions.rules file, all adding/change for mapping must be done in this file, correct?

Yes.

> In addition I need to modify gap_analysis_marking.xml to add global variables handling, built-in function processing already there, correct?

Yes.

#78 - 02/13/2017 12:50 PM - Eugenie Lyzenko

Can I add/merge your changes(with 3209e) I noted working with poc application for expressions.rules into 1521a:

```
...
     GES 20170129 Added some global variables.
         20170130 Fixed SELECTED/SELECTABLE which is a runtime stub not full support.
...
```

#79 - 02/13/2017 01:15 PM - Greg Shah

Sure. Please take all the changes to the gap analysis from there. Then remove them from 3209e so there are no merging conflicts.

#80 - 02/13/2017 01:57 PM - Eugenie Lyzenko

Greg Shah wrote:

> Sure. Please take all the changes to the gap analysis from there. Then remove them from 3209e so there are no merging conflicts.

Merged. Task branch 1521a for review updated to revision 11156.

Restoring expressions.rules in 3209e soon.

#81 - 02/13/2017 08:43 PM - Eugenie Lyzenko

Global variables sub-task

The new variables to add taken from 4GL documentation:

```
KW_CUR_LANG - CURRENT-LANGUAGE
KW_DATASRV - DATASERVERS
KW_FR_VAL - FRAME-VALUE
KW_GW - GATEWAYS
KW_GEN_PBES - GENERATE-PBE-SALT
KW_GEN_RNDK - GENERATE-RANDOM-KEY
KW_GET_CP - GET-CODEPAGES
KW_GO_PEND - GO-PENDING
KW_IS_ATTR - IS-ATTR-SPACE
KW_LASTKEY - LASTKEY or LAST-KEY
KW_MSG_LINE - MESSAGE-LINES
KW_ENTERED - ENTERED and NOT ENTERED
KW_NOW - NOW
KW_OPSYS - OPSYS
KW_OS_DRV - OS-DRIVES
KW_OS_ERR - OS-ERROR
KW_PROC_HND - PROC-HANDLE
KW_PROC_ST - PROC-STATUS
KW_PROGRESS - PROGRESS
KW_PROMSGS - PROMSGS
KW_PROPATH - PROPATH
KW_PROVER - PROVERSION
KW_RETRY - RETRY
```

```
KW_RET_VAL - RETURN-VALUE
KW_SCRN_LNS - SCREEN-LINES
KW_TERM - TERMINAL
KW_TIME - TIME
KW_TODAY - TODAY
KW_TRANS - TRANSACTION or TRANS
```

The list below is from progress.g(as alternative source to consider):

```
KW_U_CTRL
KW_U_MSG
KW_U_SERIAL
KW_U_PCTRL
KW_ACT_FORM
KW_ACT_WIN
KW_AUD_CTRL
KW_AUD_POL
KW_CLIP
KW_CODEBASE
KW_COMPILER
KW_CUR_LANG
KW_CUR_WIN
KW_DATASRV
KW_DBNAME
KW_DEBUGGER
KW_DEF_WIN
KW_ERR_STAT
KW_ETIME
KW_FIL_INFO
KW_FOCUS
KW_FR_COL
KW_FR_DB
KW_FR_DOWN
KW_FR_FIELD
KW_FR_FILE
KW_FR_INDEX
KW_FR_LINE
KW_FR_NAME
KW_FR_ROW
KW_FR_VAL
KW_GW
KW_GET_CP
KW_GO_PEND
KW_IS_ATTR
KW_LASTKEY
KW_LAST_EVT
KW_LINE_CNT
KW_LOG_MGR
KW_MSG_LINE
KW_NOW
KW_NUM_ALIA
KW_NUM_DBS
KW_OPSYS
KW_PAGE_NUM
KW_PROC_HND
KW_PROC_ST
KW_PROFILER
KW_PROGRESS
KW_PROMSGS
KW_PROPATH
KW_PROVER
KW_RCOD_INF
KW_RETRY
KW_SCRN_LNS
KW_SECUR_P
KW_SELF
KW_SESSION
KW_SUPER
KW_TERM
KW_THIS_OBJ
KW_THIS_PRC
```

```
KW_TIME
KW_TRANS
KW_USERID
```

There is variables that is not in progress.g but in 4GL documentation:

```
KW_GEN_PBES - GENERATE-PBE-SALT
KW_GEN_RNDK - GENERATE-RANDOM-KEY
KW_OS_DRV - OS-DRIVES
KW_OS_ERR - OS-ERROR
KW_RET_VAL - RETURN-VALUE
KW_TODAY - TODAY
```

Do we need them to be added into gap analysis map?

What about KW_ENTERED for ENTERED and NOT ENTERED. Can it be considered as variable to add?

#82 - 02/14/2017 08:03 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11157.

Added global variables list. Please take a look if something is wrong. The map has variables and functions without parameters(and so considering as variables).

I have added ENTERED and NOT ENTERED into the map.

The question. proc-handle and proc-status are not supported, correct?

Continue working with built-in functions.

#83 - 02/15/2017 09:19 AM - Eugenie Lyzenko

Several questions for handling built-in functions. The conversion support is performing in convert/builtin_functions.rules, correct?
I have found some entries that are not covered in convert/builtin_functions.rules:
kw_member(MEMBER) - do we have the support as declared in expressions.rules?
kw_val_evt(VALID-EVENT) - the same question(having both full support for conversion and runtime in expressions.rules).

Does it mean these functions are not supported?

The kw_p2j_rc function converting to ControlFlowOps.remoteCall is missed in expressions.rules file. Do we need it to be added?

#84 - 02/15/2017 09:59 AM - Greg Shah

Global variables are matched (see profile.rpt) using the builtin_global_variables rule from common-progress.rules. As mentioned in note 58, you should start with how we match in the reports to understand the AST node matching that will be used. This tells you where to look in the parser for the list of what the parser can produce to match this.

```
   <report
      condition="evalLib(&quot;builtin_global_variables&quot;)"
      dumpType="simple"
      multiplexExpr="execLib(&quot;describe_node&quot;, this, false)"
      prefix="builtin_global_variable_usage"
      title="Builtin Global Variable Usage"
      categories="Base Language,Expressions"/>
```

```
      <function name="builtin_global_variables">
         evalLib(&quot;variables&quot;) and
         type != prog.sys_handle       and
         !isNote("refid")              and
         parent.type != prog.object_invocation
      </function>
```

Based on this, you can see several things. For example, we do not treat system handles as global variables. The !isNote("refid") is a check to see if there is a linkage back to a local variable definition statement. If the refid exists, then this is a local variable reference and not a global variable.

When you look in progress.g, you will see the reserved_variables rule. This includes any reserved keywords that can be treated like variable references in the 4GL. But this list is not the correct list for global variables:

- it includes system handles
- it includes some keywords that will be matched as a FUNC_ token type (which is NOT a global variable, but is instead a built-in function)
- it DOES NOT include global variables that are non-reserved keywords

Please go back through your list of questions and make sure you only include VAR_ token types that are not system handles. If it is a FUNC_ type, it is NOT a global variable.

Builtin functions are converted in rules/convert/builtin_functions.rules. Global variables are converted in rules/convert/variable_references.rules.

If you follow the above checks, you will find that KW_ENTERED and NOT_ENTERED are both built-in functions that are fully supported.

You will also see that things like KW_DEBUGGER, KW_PROC_HND, KW_PROC_ST, KW_PROFILER and so forth are system handles NOT global variables.

Please look in the parser at calls to addGlobalVariable() and exclude the ones that are FUNC_ types or of type SYS_HANDLE.

#85 - 02/15/2017 10:10 AM - Eugenie Lyzenko

> Please go back through your list of questions and make sure you only include VAR_ token types that are not system handles. If it is a FUNC_ type, it is NOT a global variable.

OK. But what about functions that might not have the parameters and can be used without parenthesis? Shouldn't they be considered as global variables?

#86 - 02/15/2017 10:58 AM - Greg Shah

If they have FUNC_ types they are not treated as global variables.

#87 - 02/15/2017 04:57 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11158.

Fix removing function and system handles from global variable map. According to all criteria SUPER(kw_super) and THIS-OBJECT(kw_this_obj) are not global variables, removed too.

#88 - 02/15/2017 07:44 PM - Eugenie Lyzenko

OK. According to my refresh for variable/built-in functions criteria the SUPER(kw_super) is the function, not variable.

But not clear about _msg(kw_u_msg). In progress.g the addGlobalVariable() defines it as VAR_INT but expressions.rules has it inside addBuiltinFuncs map definition. This means it can be the global variable and builtin function simultaneously?

#89 - 02/15/2017 08:41 PM - Greg Shah

Good catch. I think this:

```
   sym.addGlobalVariable("_msg"                , VAR_INT           );
```

should actually be this:

```
   sym.addGlobalVariable("_msg"                , FUNC_INT          );
```

The inclusion in expressions.rules as a built-in function is correct.

#90 - 02/15/2017 08:48 PM - Greg Shah

Please fix progress.g for _msg.

#91 - 02/15/2017 09:17 PM - Eugenie Lyzenko

Greg Shah wrote:

> Please fix progress.g for _msg.

OK. Task branch 1521a for review updated to revision 11159, 11160(javadoc completion).

Fix for _msg function. Removed from expressions.rules global variable map and the progress.g has been changed to properly map the _msg to FUNC_INT.

#92 - 02/15/2017 10:33 PM - Eugenie Lyzenko

The new candidates for built-in functions to add into expressions.rules map:

```
"audit-enabled"        , FUNC_LOGICAL    , false);
"box"                  , FUNC_CLASS      , true , "System.Object"); // TODO: this qualified class is wrong but
 we need a placeholder
"cast"                 , FUNC_CLASS      , true ); // used for annotations only
"connected"            , FUNC_LOGICAL    , true ); is kw_conn_ed
"data-source-modified" , FUNC_LOGICAL    , false);
"dbcodepage"           , FUNC_CHAR       , true );
"dbcollation"          , FUNC_CHAR       , true );
"db-remote-host"       , FUNC_CHAR       , true );
"dbtaskid"             , FUNC_INT        , true );
? "decimal"              , FUNC_DEC        , true );
? "decrypt"              , FUNC_MEMPTR     , false);
"dynamic-cast"         , FUNC_CLASS      , true , "Progress.Lang.Object");  // TODO: this qualified class is w
rong but we need a placeholder
"dynamic-invoke"       , FUNC_POLY       , true );
"get-collation"        , FUNC_CHAR       , false);
"get-collations"       , FUNC_CHAR       , true );
"guid"                 , FUNC_CHAR       , false);
"is-column-codepage"   , FUNC_LOGICAL    , false);
"is-lead-byte"         , FUNC_LOGICAL    , true );
"library"              , FUNC_CHAR       , true );
"list-events"          , FUNC_CHAR       , true );
"list-query-attrs"     , FUNC_CHAR       , true );
"list-set-attrs"       , FUNC_CHAR       , true );
"list-widgets"         , FUNC_CHAR       , true );
"load-picture"         , FUNC_COM_HANDLE , true ); // supposed to be a "statement" but really acts like a func
tion
"lower"                , FUNC_CHAR       , true );
"p2j-remote-call"      , FUNC_CHAR       , true); // FWD-extension, not real 4GL!
"raw"                  , FUNC_RAW        , false);
"record-length"        , FUNC_INT        , false);
"rejected"             , FUNC_LOGICAL    , false);
"row-state"            , FUNC_INT        , false);
"set-db-client"        , FUNC_LOGICAL    , false);
"ssl-server-name"      , FUNC_CHAR       , false);
"type-of"              , FUNC_LOGICAL    , false);
"upper"                , FUNC_CHAR       , true );
"user"                 , FUNC_CHAR       , false);
"valid-object"         , FUNC_LOGICAL    , false);
```

Not clear for now what to do with "webspeed" functions.

#93 - 02/16/2017 05:03 AM - Greg Shah

Yes, please go ahead with adding all of these, including the webspeed functions. Treat the webspeed functions just like other built-ins. Currently we have no conversion or runtime support for the webspeed functions.

Also: please look at 3209e and ensure that the 1521a support levels are matching what is available in 3209e. We are trying to get 3209e into the trunk ASAP and 1521a will follow that. I think we have added some support in 3209e that probably is not represented in 1521a yet.

#94 - 02/16/2017 12:18 PM - Eugenie Lyzenko

Greg Shah wrote:

> Yes, please go ahead with adding all of these, including the webspeed functions. Treat the webspeed functions just like other built-ins. Currently we have no conversion or runtime support for the webspeed functions.

OK.

> Also: please look at 3209e and ensure that the 1521a support levels are matching what is available in 3209e. We are trying to get 3209e into the trunk ASAP and 1521a will follow that. I think we have added some support in 3209e that probably is not represented in 1521a yet.

I have merged rule files related changes in 1521a. And I have reviewed the changes in 3209e since this merge. Do you think something is missed?

Another point. Please consider the following keywords in progress.g(line 29427):

```
...
         new Keyword("hidden-field"                    ,  0, KW_HID_FLD , false),  // not a real keyword, but i
s normally defined in the PSC webspeed 4GL code
         new Keyword("hidden-field-list"               ,  0, KW_HID_FLD , false),  // not a real keyword, but i
s normally defined in the PSC webspeed 4GL code
...
```

Looks like there is an error here, the keyword value for "hidden-field-list" must be KW_HID_FLDL, right? Like for the "url-field"/"url-field-list" pair.

```
...
         new Keyword("hidden-field-list"               ,  0, KW_HID_FLDL, false),  // not a real keyword, but i
s normally defined in the PSC webspeed 4GL code
...
```

#95 - 02/16/2017 12:21 PM - Greg Shah

> And I have reviewed the changes in 3209e since this merge. Do you think something is missed?

Yes, I think there were many changes in this branch that did not already have matching updates to expressions.rules. Please review the entire branch for changes.

> Looks like there is an error here, the keyword value for "hidden-field-list" must be KW_HID_FLDL, right?

Yes, please fix it.

#96 - 02/16/2017 01:09 PM - Eugenie Lyzenko

Greg Shah wrote:

> > And I have reviewed the changes in 3209e since this merge. Do you think something is missed?
>
> Yes, I think there were many changes in this branch that did not already have matching updates to expressions.rules. Please review the entire branch for changes.

OK. I will re-check the built-in functions and global variables with not full support status for the changes. After completing the new built-in functions adding. After verifying the status of the new functions to be exact.

#97 - 02/16/2017 03:47 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11161.

Preparing candidate for global variables and built-in functions. Fixed wrong keyword constant in progress.g.

Continue with integrating report generation into profile.rpt.

#98 - 02/16/2017 07:42 PM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11162.

In this update the global variables and built-in functions report generation is working. The gap_analysis_marking.xml was modified to include the logic to handle global variables. I have tested the change with POC application and report is OK.

So until it will be discovered the missing functions and variables the subtask is ready. I could test it within another bigger source project.

Another point I have noted working with gap analysis is the rule logic schema in gap_analysis_marking.xml. Consider this:

```
...
         <!-- built-in functions (excluding those that act like global vars) -->
         <rule>evalLib("builtin_funcs")
            <action>mark = (rw.cvt_lvl_none | rw.rt_lvl_none)</action>
            <rule>funcs.containsKey(otype)
               <action>mark = funcs.get(otype)</action>
            </rule>
         </rule>

         <!-- attributes/methods -->
         <rule>evalLib("methods_and_attributes")
            <action>mark = (rw.cvt_lvl_none | rw.rt_lvl_none)</action>
            <rule>attrmeth.containsKey(otype)
               <action>mark = attrmeth.get(otype)</action>
            </rule>
         </rule>

         <!-- global variables -->
         <rule>evalLib("builtin_global_variables")
            <action>mark = (rw.cvt_lvl_none | rw.rt_lvl_none)</action>
            <rule>globalVars.containsKey(otype)
               <action>mark = globalVars.get(otype)</action>
            </rule>
         </rule>
...
```

I see here something I would changed:
1. If the mark has been found on some step -> we can ignore further steps, saving CPU time, like this:

```
...
         <!-- built-in functions (excluding those that act like global vars) -->
         <rule>evalLib("builtin_funcs")
            <action>mark = (rw.cvt_lvl_none | rw.rt_lvl_none)</action>
            <rule>funcs.containsKey(otype)
               <action>mark = funcs.get(otype)</action>
            </rule>
         </rule>

         <!-- attributes/methods -->
         <rule>mark == rw.lvl_unknown and evalLib("methods_and_attributes")
            <rule>attrmeth.containsKey(otype)
               <action>mark = attrmeth.get(otype)</action>
            </rule>
         </rule>
...
```

2. On the other hand the current approach assumes if some evalLib("criteria") is match all previous values for mark variable will be overridden. Does it mean we can lost some info if we already have correct mark value? Or we are preserved by the fact the keyword(element) can not be member of the multiple maps simultaneously?

#99 - 02/17/2017 06:12 AM - Greg Shah

> If the mark has been found on some step -> we can ignore further steps, saving CPU time, like this:

Good idea.

> On the other hand the current approach assumes if some evalLib("criteria") is match all previous values for mark variable will be overridden. Does it mean we can lost some info if we already have correct mark value?

No, this would not be intended.

> Or we are preserved by the fact the keyword(element) can not be member of the multiple maps simultaneously?

The intention is for each case to be mutually exclusive.

#100 - 02/17/2017 08:29 AM - Eugenie Lyzenko

Task branch 1521a for review updated to revision 11163.

Adding changes in gap_analysis_marking.xml to:
1. Eliminate processing when keyword match has already been found.
2. Removed code:

```
            <action>mark = (rw.cvt_lvl_none | rw.rt_lvl_none)</action>
```

This can help to verify the case when we match criteria but have missing entry in our status maps(will be shown with unknown status instead of none). We can even introduce some special constant for such cases, say rw.lvl_todo to separate it from rw.lvl_unknown and from (rw.cvt_lvl_none | rw.rt_lvl_none) so the rule can be substituted with:

```
            <action>mark = rw.lvl_todo</action>
```

as default value.

The point 2 can be restored further when we are sure all status maps are 100% completed. I think we need to preform additional testing with some big enough 4GL code.

#101 - 03/04/2017 10:34 AM - Greg Shah

The latest version of 1521a is 11164. I have merged this into 3255a. Please do NOT do any additional development on 1521a. As soon as you confirm that any pending work (uncommitted changes) in 1521a have been made safe and applied to 3255a, we will archive 1521a.

I needed to base my recent work on #3255 on the gap analysis improvements in 1521a. From there Eric and I both added many fixes. That is why 3255a is the "going forward" branch for both tasks.

#102 - 03/04/2017 12:09 PM - Eugenie Lyzenko

Greg Shah wrote:

> The latest version of 1521a is 11164. I have merged this into 3255a. Please do NOT do any additional development on 1521a. As soon as you confirm that any pending work (uncommitted changes) in 1521a have been made safe and applied to 3255a, we will archive 1521a.
>
> I needed to base my recent work on #3255 on the gap analysis improvements in 1521a. From there Eric and I both added many fixes. That is why 3255a is the "going forward" branch for both tasks.

OK.

#103 - 03/06/2017 09:24 AM - Eugenie Lyzenko

Greg Shah wrote:

> The latest version of 1521a is 11164. I have merged this into 3255a. Please do NOT do any additional development on 1521a. As soon as you confirm that any pending work (uncommitted changes) in 1521a have been made safe and applied to 3255a, we will archive 1521a.

OK. There is no my changes to commit over current 1521a. The repo is in sync with my local development, I guess we can archive 1521a.

#104 - 03/07/2017 07:18 AM - Greg Shah

1521a has been archived.

3255a contains a large number of improvements including some cleanup of the frame/browse/widget options (and the matching reports). Among other changes, I made these sections mutually exclusive. Otherwise the results were confusing and in some cases incorrect.

Stanislav: would you please review the browse-related values? I have some SVL: comments there about some things, but all of it is worthy of review. I'm especially concerned that we may be missing some options which previously we got market by the widget rules. Of course, if the option really is not possible for browse then it is not a concern. I'm just worried that I may have missed something.

#105 - 03/07/2017 07:27 AM - Greg Shah

- % Done changed from 20 to 0

Eugenie: your next task is to encode marking rules for the following:

- language statements (we just need to mark at the top level so that the language statement reports can be enabled for gap analysis)
- i/o options

#106 - 03/07/2017 07:33 AM - Eugenie Lyzenko

Greg Shah wrote:

> Eugenie: your next task is to encode marking rules for the following:
>
> - language statements (we just need to mark at the top level so that the language statement reports can be enabled for gap analysis)
> - i/o options

OK.

Do I need to create 1521b branch for this task? Or this work will be inside 3255a?

#107 - 03/07/2017 07:40 AM - Greg Shah

Put the changes in 3255a. Please make sure you don't break anything with your check-ins, because others are working in this branch too.

#108 - 03/07/2017 08:08 AM - Eugenie Lyzenko

Greg Shah wrote:

> Put the changes in 3255a. Please make sure you don't break anything with your check-ins, because others are working in this branch too.

OK. Who will make rebase the branch with new trunk? We need to sync this process to avoid 3255a corruptions.

#109 - 03/07/2017 08:35 AM - Greg Shah

I will do this shortly.

#110 - 03/07/2017 11:02 AM - Eugenie Lyzenko

Greg Shah wrote:

> Eugenie: your next task is to encode marking rules for the following:

- language statements (we just need to mark at the top level so that the language statement reports can be enabled for gap analysis)

The plan is to:

1. Add new function addLangStatemts to the user_interface.rules or may be create separate *.rules file for this if language statement are not only user interface related.
2. Modify gap_analysis_marking.xml to add calculation of the mark level.

#111 - 03/07/2017 11:13 AM - Greg Shah

Please create lang_stmts.rules. Language statements cover all areas.

To see the matching criteria, look at the statements function in common-progress.rules.

#112 - 03/07/2017 11:26 AM - Greg Shah

Consider 3255a locked. I am rebasing it right now.

#113 - 03/07/2017 11:56 AM - Greg Shah

3255a is now based on trunk revision 11141. I could not use the normal rebase process since this branch was composed of 3209e (rev 11222), 1521a and many additional changes.

Instead, I took a fresh copy of trunk and merged thye old 3255a revs 11123 through 11138 into it. The new 3255a has its latest revision as 11142. If you want to see the history, use bzr log -v -n0.

#114 - 03/08/2017 05:04 PM - Eugenie Lyzenko

Task branch 3255a for review updated to revision 11143.

This is the first step for iterative creation of the language statements gap analysis map. Includes the keywords that directly mentioned in common-progress.rules. Planning to search progress.g file for more statements and look at official 4GL documentation to double-check.

For now I have found the statement DYNAMIC-PROPERTY does not even have the keyword in progress.g to refer to. Is it expected, or we need to add something like kw_dyn_prop as keyword?

I have added runtime support for SET-DOUBLE as rt_lvl_full_restr because we use use integer data worker to handle double data type(the same is for SET-FLOAT). Is it OK or the support should be mentioned as rt_lvl_full?

Continue working.

#115 - 03/09/2017 01:51 PM - Greg Shah

Code Review Task Branch 1521a Revision 11143

1. put-bits is fully supported. It is not part of memptr. It is for numbers (usually integers). The public API is in NumberType and the actual implementation occurs in the abstract method setBitsWorker() which is handled by subclasses.

2. The kw_put_dbl and kw_put_flt are fully supported. There is a TODO that we want to confirm that our encoding of the bits is exactly correct. Please code this as "rt_lvl_basic". The issue has nothing to do with the use of integer for writing it. The question relates to how we encode the floating point bits into a bit field for insertion into memory.

3. In regard to dynamic-property, ignore it. It is not supported in the parser yet because it is a recent addition.

#116 - 03/09/2017 09:41 PM - Eugenie Lyzenko

Task branch 3255a for review updated to revision 11144.

Importing language statements from progress.g file. And notes resolution. Not completed - the TODO items commented out.

Some questions:
1. I have not found KW_BLK_LVL keyword(block_level_stmt in progress.g). This keyword does not have the statement?
2. The Host Language Call statement(KW_CALL) is not supported?
3. Do we need the DB related statements to be included in lang_stmt.rules? Or all DB related will be processed in separate database.rules file.

Continue working.

#117 - 03/10/2017 05:38 AM - Greg Shah

> I have not found KW_BLK_LVL keyword(block_level_stmt in progress.g). This keyword does not have the statement?

stmt_list calls block_level_stmt which is KW_BLK_LVL^ on_event_phrase DOT!

stmt_list and assign_type_syntax_stmt_list are the places you should look for statements.

> The Host Language Call statement(KW_CALL) is not supported?

Correct.

> Do we need the DB related statements to be included in lang_stmt.rules? Or all DB related will be processed in separate database.rules file.

Keep it in one lang_stmt.rules file for now.

#118 - 03/10/2017 09:58 PM - Eugenie Lyzenko

Task branch 3255a for review updated to revision 11147.

Finishing import the language statements from progress.g. Also notes resolution(I saw the repo was already fixed within 11146).

Not very clear what is the status of the SQL direct statements, I have marked them as not supported. Also not found the support for the INTERFACE(KW_INTERFAC) and METHOD(KW_METHOD) statements.

Continue working.

#119 - 03/13/2017 09:24 AM - Greg Shah

Code Review Task Branch 3255a Revision 11147

This was a good start, but there were many errors. Some problems were with the marked levels. But some issues were related to a misunderstanding about the token type being matched. I have fixed these problems and checked in rev 11148.

There are many places in the parser where we match on a token (or multiple tokens) and then change the token type to an artificial type which is not ambiguous.

KW_PROCESS KW_EVENTS is PROCESS_EVENTS
KW_INPUT KW_CLEAR is INPUT_CLEAR
KW_INPUT KW_THROUGH is INPUT_THRU
...

We also have procedure, function, kw_class, kw_method, kw_interface, kw_construc, kw_destruct defined in the list, but we don't match on these as "language statements". We probably should change the matching.

I need to get this branch into the trunk. I do want to honor the language statements rules in the reports and test this to make sure it is safe. Please do NOT make any changes in this branch at this time, it is FROZEN.

#120 - 03/13/2017 09:46 AM - Eugenie Lyzenko

Greg Shah wrote:

> Code Review Task Branch 3255a Revision 11147
>
> This was a good start, but there were many errors. Some problems were with the marked levels. But some issues were related to a misunderstanding about the token type being matched. I have fixed these problems and checked in rev 11148.
>
> There are many places in the parser where we match on a token (or multiple tokens) and then change the token type to an artificial type which is not ambiguous.
>
> KW_PROCESS KW_EVENTS is PROCESS_EVENTS
> KW_INPUT KW_CLEAR is INPUT_CLEAR
> KW_INPUT KW_THROUGH is INPUT_THRU
> ...

You mean for example the cases like:

```
DEFINE BUTTON
DEFINE VARIABLE
...
```

We need to include only:

```
        <rule>opts.put(prog.define_button,              rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
        <rule>opts.put(prog.define_variable,            rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
```

And **NOT** to include:

```
        <rule>opts.put(prog.kw_define,                  rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
```

becuse the word DEFINE is not a statements itself without object we are going to define. Right?

> We also have procedure, function, kw_class, kw_method, kw_interface, kw_construc, kw_destruct defined in the list, but we don't match on these as "language statements". We probably should change the matching.

I have added them based on 4GL ABL Reference book declaration as statement. But agree we can consider with another meaning from conversion point of view.

> I need to get this branch into the trunk. I do want to honor the language statements rules in the reports and test this to make sure it is safe. Please do NOT make any changes in this branch at this time, it is FROZEN.

OK.

#121 - 03/13/2017 10:47 AM - Greg Shah

> You mean for example the cases like:
> ...
> We need to include only:

...

> And NOT to include:

...

> becuse the word DEFINE is not a statements itself without object we are going to define. Right?

Exactly.

> I have added them based on 4GL ABL Reference book declaration as statement. But agree we can consider with another meaning from conversion point of view.

I understand. This was broadly correct. The only problem is that the "statements" function will never match them. I did add a new control_flow.rules and marking for blocks. So now these are covered too.

#122 - 03/13/2017 11:07 AM - Greg Shah

3255a has been "rebased" from trunk 11142. The rebase process is broken for this branch so I had to merge. The latest revision is 11143.

#123 - 03/13/2017 12:12 PM - Eugenie Lyzenko

Greg Shah wrote:

> 3255a has been "rebased" from trunk 11142. The rebase process is broken for this branch so I had to merge.

Is this caused by my commits: 11143, 11144, 11147?

#124 - 03/13/2017 12:20 PM - Greg Shah

No, this is because we built this from a combination of 1521a and 3209e. Then a later version of 3209e was merged to trunk. Rebasing was broken because of that, I think.

#125 - 03/13/2017 03:21 PM - Greg Shah

3255a is locked for regression testing. I'm going to test using rev 11145.

#126 - 03/14/2017 05:59 AM - Greg Shah

3255a branch revision 11145 passed ChUI regression testing. The conversion result was identical with trunk. The runtime regression passed on the first run (it only had the expected failure at TC_JOB_002 step 40).

Eric is doing regression testing with ETF. If that passes we will merge this to trunk today. The branch is still locked.

#127 - 03/14/2017 09:19 AM - Greg Shah

ETF testing passed. Re-run of reporting was successful. Testing is complete.

3255a has been merged to trunk as revision 11143.

#128 - 03/14/2017 09:20 AM - Greg Shah

I've created task branch 1521b from trunk 11143.

#129 - 03/14/2017 11:10 AM - Eugenie Lyzenko

Greg Shah wrote:

> I've created task branch 1521b from trunk 11143.

So any further updated for this task will be into 1521b, correct?

#130 - 03/14/2017 11:17 AM - Greg Shah

Yes.

I'm doing the final cleanup on language statements. Consider that complete.

I'll let you know the next priorities soon.

#131 - 03/14/2017 11:18 AM - Eugenie Lyzenko

Greg Shah wrote:

> Yes.
>
> I'm doing the final cleanup on language statements. Consider that complete.
>
> I'll let you know the next priorities soon.

OK.

#132 - 03/20/2017 10:24 PM - Eugenie Lyzenko

Studied the task objective. Correct me if I'm wrong. I have scanned convert/expression.rules and found the candidates for EXPRESSION related:

```
prog.kw_while
prog.kw_when
prog.kw_to
prog.kw_by
prog.query_subst
```

I have a doubt if my criteria is correct because they are parent to the prog.expression

For the ASSIGN statement planning to scan in progress.g with items from assign_type_syntax_stmt_list

```
current_value_stmt
dynamic_current_value_stmt
dynamic_new_stmt
entry_stmt
extent_stmt
fix_codepage_stmt
length_stmt
overlay_stmt
put_memptr_stmt
raw_stmt
set_byte_order_stmt
set_pointer_value_stmt
set_size_stmt
substring_stmt
```

to find a keyword which are not already being marked.

#133 - 03/21/2017 06:51 AM - Greg Shah

I want you to focus on the expr rule in progress.g. Do not worry about convert/expression.rules.

> I have a doubt if my criteria is correct because they are parent to the prog.expression

These are not correct. Ignore them.

> For the ASSIGN statement planning to scan in progress.g with items from assign_type_syntax_stmt_list

We have already handled these in lang_stmts.rules. Ignore them.

Focus on expr.

#134 - 03/21/2017 10:01 PM - Eugenie Lyzenko

The keywords extracted from different kind of expressions(in progress.g):

```
DIVIDE
EQUALS
GT
GTE
IS_NOT_NULL
IS_NULL
KW_AND
KW_BEGINS
KW_BETWEEN
KW_CONTAINS
KW_IN
KW_IS
KW_LIKE
KW_MATCHES
KW_MOD
KW_NOT
KW_OR
LT
LTE
MULTIPLY
NOT_BETWEEN
NOT_EQ
NOT_IN
NOT_LIKE
SYMBOL(for DB_REF_NON_STATIC)
UN_MINUS
UN_PLUS
```

The SYMBOL can contain sub keyword may be? Like / as DIVIDE or SLASH?

I think all of them can be grouped into associated language elements, requires one or more items to consider with. Like a "pretext" or "union" elements in verb language not having any meaning by itself(without items these keywords are associated with).

#135 - 03/22/2017 09:02 AM - Greg Shah

Most of the token types in your list are already handled by the operators list in gaps/expressions.rules, right?

> The SYMBOL can contain sub keyword may be? Like / as DIVIDE or SLASH?

Only at the parser level, not in the AST. It is "put back together" and the token type is set to SYMBOL.

#136 - 03/22/2017 05:50 PM - Eugenie Lyzenko

Greg Shah wrote:

> Most of the token types in your list are already handled by the operators list in gaps/expressions.rules, right?

Removed already handled keywords:

```
COM_INVOCATION
IS_NOT_NULL
IS_NULL
KW_BETWEEN
KW_IN
KW_IS
KW_LIKE
NOT_BETWEEN
NOT_IN
NOT_LIKE
SYMBOL(for DB_REF_NON_STATIC)
```

What about 9th and highest precendence level of Progress 4GL expressions? Do we need to consider the cases included in primary_expr?

```
literal
new_phrase
sql_aggregate_funcs
...
```

#137 - 03/23/2017 07:38 AM - Greg Shah

> What about 9th and highest precendence level of Progress 4GL expressions? Do we need to consider the cases included in primary_expr?

Yes.

Do remember that some of these may already be addressed in gaps/expressions.rules. For example, check the literals and the built-in functions (for the sql_aggregate_funcs).

#138 - 03/23/2017 01:29 PM - Greg Shah

1521b is frozen at this time. Eric is putting it through conversion regression testing. It has no runtime implications. If conversion testing passes it will be merged to trunk.

#139 - 03/23/2017 04:05 PM - Eugenie Lyzenko

Greg,

Checking the keywords for already handled status I found small issue in gaps/expression.rules file:

```
...
         <rule>funcs.put(prog.kw_compare , rw.cvt_lvl_full    | rw.rt_lvl_partial)</rule>
         <rule>funcs.put(prog.kw_conn_ed , rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
         <rule>funcs.put(prog.kw_conn_ed , rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
...
```

Line 138, the keyword kw_conn_ed is defined twice. I guess one line should be removed.

#140 - 03/23/2017 04:17 PM - Greg Shah

Fixed.

#141 - 03/23/2017 04:57 PM - Hynek Cihlar

I checked in "Made 'all' build target depend on 'distribution' target." to 1521b revision 11149.

#142 - 03/24/2017 12:25 AM - Eugenie Lyzenko

New expression related(direct or indirect from 9th and highest precedence level of Progress 4GL expressions) keywords not yet handled:

```
CLASS_EVENT
CLASS_NAME
DB_SYMBOL
```

FIELD_BLOB
FIELD_CHAR
FIELD_CLASS
FIELD_CLOB
FIELD_COM_HANDLE
FIELD_DATE
FIELD_DATETIME
FIELD_DATETIME_TZ
FIELD_DEC
FIELD_HANDLE
FIELD_INT
FIELD_INT64
FIELD_LOGICAL
FIELD_RAW
FIELD_RECID
FIELD_ROWID
FUNC_CHAR
FUNC_CLASS
FUNC_COM_HANDLE
FUNC_DATE
FUNC_DATETIME
FUNC_DATETIME_TZ
FUNC_DEC
FUNC_HANDLE
FUNC_INT
FUNC_INT64
FUNC_LOGICAL
FUNC_LONGCHAR
FUNC_MEMPTR
FUNC_POLY
FUNC_RAW
FUNC_RECID
FUNC_ROWID
KW_BROWSE
KW_DATA_REL
KW_DSET_HND
KW_ELSE
KW_STRM_HND
KW_TEMP_TAB
KW_THEN
METH_CHAR
METH_CLASS
METH_COM_HANDLE
METH_DATE
METH_DATETIME
METH_DATETIME_TZ
METH_DEC
METH_HANDLE
METH_INT
METH_INT64
METH_LOGICAL
METH_LONGCHAR
METH_MEMPTR
METH_POLY
METH_RAW
METH_RECID
METH_ROWID
METH_VOID
OBJECT_INVOCATION
SYS_HANDLE
UNKNOWN_TOKEN
VAR_BYTE
VAR_CHAR
VAR_CLASS
VAR_COM_HANDLE
VAR_DATE
VAR_DATETIME
VAR_DATETIME_TZ
VAR_DEC
VAR_DOUBLE
VAR_FLOAT
VAR_HANDLE
VAR_INT
VAR_INT64
VAR_LOGICAL

```
VAR_LONG
VAR_LONGCHAR
VAR_MEMPTR
VAR_RAW
VAR_RECID
VAR_ROWID
VAR_SHORT
VAR_USHORT
WID_BROWSE
WID_BUTTON
WID_COMBO
WID_DIALOG
WID_EDITOR
WID_FILL_IN
WID_FRAME
WID_FRAME
WID_IMAGE
WID_LITERAL
WID_MENU
WID_MENU_ITM
WID_RADIO
WID_RECT
WID_SEL_LST
WID_SLIDER
WID_SUB_MENU
WID_TEXT
WID_TOGGLE
WID_WINDOW
```

General gap concept question. We need to enumerate all possible keywords and put them in a separate groups by some criteria. Is this correct understanding?

#143 - 03/24/2017 08:24 AM - Greg Shah

> General gap concept question. We need to enumerate all possible keywords and put them in a separate groups by some criteria. Is this correct understanding?

Yes.

By the way, we already handle all the FUNC_ types that are marked as "builtin". The non-builtin function calls still would need to be marked.

#144 - 03/24/2017 10:54 AM - Eugenie Lyzenko

Greg Shah wrote:

> By the way, we already handle all the FUNC_ types that are marked as "builtin". The non-builtin function calls still would need to be marked.

The remaining FUNC_ that are not marked are:

```
FUNC_CHAR
FUNC_CLASS (not sure, control flow gap keyword may be)
FUNC_COM_HANDLE
FUNC_LONGCHAR
FUNC_MEMPTR
FUNC_POLY
FUNC_ROWID
```

Correct?

There are two more double items found for built-in functions:

```
...
         <rule>funcs.put(prog.kw_caps    , rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
         <rule>funcs.put(prog.kw_caps    , rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
...
         <rule>funcs.put(prog.kw_dec     , rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
         <rule>funcs.put(prog.kw_dec     , rw.cvt_lvl_full    | rw.rt_lvl_full)</rule>
...
```

#145 - 03/24/2017 11:00 AM - Greg Shah

> The remaining FUNC_ that are not marked are:
> ...
> Correct?

No. We already mark any FUNC_ type that is marked as a "builtin". This means it has an boolean annotation of that name. All FUNC_ nodes that DO NOT have that annotation are NOT marked. Your job is to implement a way to mark the builtin == false case. Please look carefully at builtin_funcs and user_funcs in common-progress.rules.

It is important for you to look carefully at the conditions we use for marking in gap_analysis_marking.xml so that you can understand exactly what is and what is not marked.

#146 - 03/24/2017 11:00 AM - Greg Shah

> There are two more double items found for built-in functions:

Fix those in 1521c.

#147 - 03/24/2017 03:03 PM - Eugenie Lyzenko

Created task branch 1521c from trunk revision 11144.

#148 - 03/24/2017 04:38 PM - Eugenie Lyzenko

Greg Shah wrote:

> > There are two more double items found for built-in functions:
>
> Fix those in 1521c.

Done. Task branch 1521c for review updated to revision 11145.

#149 - 03/27/2017 09:11 AM - Eugenie Lyzenko

The candidate list for expression related keywords to add:

```
ATTR_CLASS
BOOL_FALSE (SYMBOL, marked as known literal)
BOOL_TRUE (SYMBOL, marked as known literal)
BUFFER
CLASS_EVENT
CLASS_NAME
COLON
COM_INVOCATION
COM_METHOD
COM_PROPERTY
DB_SYMBOL (SYMBOL)
FIELD_BLOB
FIELD_CHAR
FIELD_CLASS
FIELD_CLOB
FIELD_COM_HANDLE
FIELD_DATE
FIELD_DATETIME
FIELD_DATETIME_TZ
FIELD_DEC
FIELD_HANDLE
FIELD_INT
FIELD_INT64
FIELD_LOGICAL
FIELD_RAW
FIELD_RECID
FIELD_ROWID
FILENAME (SYMBOL)
IS_NOT_NULL
IS_NULL
KW_ALL
KW_ANY
KW_AS
KW_BETWEEN
KW_BROWSE
KW_BY_PTR
KW_BY_VAR_P
KW_DATA_REL
KW_DISTINCT
KW_DSET_HND
KW_ESCAPE
KW_EXISTS
KW_FIND_CS
KW_FIND_GLO
KW_FIND_NO
KW_FIND_PO
KW_FIND_WA
KW_FROM
KW_IN
KW_IS
KW_ROW_CRTD
KW_ROW_DELD
KW_ROW_MODD
KW_ROW_UMOD
KW_SELECT
KW_SOME
KW_STRM_HND
KW_WIN_DMIN
KW_WIN_MAX
KW_WIN_MIN
KW_WIN_NORM
METH_CHAR
METH_CLASS
METH_COM_HANDLE
METH_DATE
METH_DATETIME
METH_DATETIME_TZ
METH_DEC
METH_HANDLE
METH_INT
```

```
METH_INT64
METH_LOGICAL
METH_LONGCHAR
METH_MEMPTR
METH_POLY
METH_RAW
METH_RECID
METH_ROWID
METH_VOID
NOT_BETWEEN
NOT_EXISTS
NOT_IN
NOT_LIKE
OBJECT_INVOCATION
QUERY
STREAM
STRING
SYMBOL
SYS_HANDLE
TABLE
TEMP_TABLE
UNKNOWN_VAL (SYMBOL, marked as known literal)
VAR_BYTE
VAR_CHAR
VAR_CLASS
VAR_COM_HANDLE
VAR_DATE
VAR_DATETIME
VAR_DATETIME_TZ
VAR_DEC
VAR_DOUBLE
VAR_FLOAT
VAR_HANDLE
VAR_INT
VAR_INT64
VAR_LOGICAL
VAR_LONG
VAR_LONGCHAR
VAR_MEMPTR
VAR_RAW
VAR_RECID
VAR_ROWID
VAR_SHORT
VAR_USHORT
WID_BROWSE
WID_BUTTON
WID_COMBO
WID_DIALOG
WID_EDITOR
WID_FILL_IN
WID_FRAME
WID_IMAGE
WID_LITERAL
WID_MENU
WID_MENU_ITM
WID_RADIO
WID_RECT
WID_SEL_LST
WID_SLIDER
WID_SUB_MENU
WID_TEXT
WID_TOGGLE
WID_WINDOW
WORK_TABLE
```

The question for UNKNOWN_TOKEN. This special artificial keyword means the detected keyword behind is not registered in conversion system. So it will not be presented anywhere in gap analysis, right?

The second question for keywords KW_ELSE and KW_THEN. The keyword KW_IF already marked in lang_stmts.rules. Do we need to add these 2 keywords there too? Because all of them belong to the same IF .. THEN .. ELSE language statement.

For the rest of the keywords mentioned above. I'm planning to make new map inside gaps/gap_analysis_marking.xml, say exprChains and define the

function to add them within gaps/expressions.rules file. Is it OK? Or may be we need to separate this list by meaning, some of them are widget related, some of them are VAR_* related, ...? I just afraid we will have a lot of groups with small entries inside as result of this separation.

#150 - 03/27/2017 05:04 PM - Eugenie Lyzenko

I have checked once again all already marked categories in gaps/gap_analysis_marking.xml and have not found something is already marked. I think all of them can be a part of the expression(in terms of language sequence that must be evaluated) directly like X + Y or indirectly like X + object:property value. The exception is probably KW_BROWSE as properly marked in gaps/lang_stmts.rule file with map addLangStatements and should be removed from the list.

#151 - 03/28/2017 05:34 AM - Greg Shah

> The question for UNKNOWN_TOKEN. This special artificial keyword means the detected keyword behind is not registered in conversion system. So it will not be presented anywhere in gap analysis, right?

Yes. In all the places of the parser where we match on UNKNOWN_TOKEN, we rewrite the token type to some other value. For this reason, the UNKNOWN_TOKEN will never be seen in downstream conversion code.

> The second question for keywords KW_ELSE and KW_THEN. The keyword KW_IF already marked in lang_stmts.rules. Do we need to add these 2 keywords there too? Because all of them belong to the same IF .. THEN .. ELSE language statement.

No, KW_ELSE and KW_THEN do not need to be marked. However, you are misunderstanding something. The ABL has two forms of the IF, one is a language statement and the other is the IF "built-in function" that operates like the Java ternary operator. It is this IF/THEN/ELSE that will be matched from an expression:

```
if_func
   :
      i:KW_IF^ { saveAndReplaceType(#i, FUNC_POLY); }
      expr KW_THEN! expr KW_ELSE! expr
```

But please notice that we rewrite the token type to FUNC_POLY and this will be marked as a built-in. So the KW_IF is already handled. And also notice the ! after the KW_THEN and KW_ELSE. This means that we match these tokens and drop these nodes from the AST. They never appear in downstream conversion code, so they don't have to be marked.

> to make new map inside gaps/gap_analysis_marking.xml, say exprChains and define the function to add them within gaps/expressions.rules file. Is it OK?

No, I don't want to have unrelated processing marked by the same map. Remember the concept here is to mark using very specific matching rules. So the funcs map in gap_analysis_marking.xml is only accessed when we match evalLib("builtin_funcs"). This means that the specific funcs map is for matching builtin functions only. It makes it safe to have some of those same keywords appear in other maps (e.g. language statements) because those entries are protected by matching logic too. Since the ABL has so many ambiguous keywords (keywords used for multiple purposes), we MUST separate everything out into the specific use-cases.

Doing this separation also helps us find and maintain things more easily since they are grouped logically.

> Or may be we need to separate this list by meaning, some of them are widget related, some of them are VAR_* related, ...? I just afraid we will have a lot of groups with small entries inside as result of this separation.

Yes, they must be separated. It is OK for there to be small groups.

I see some obvious groups:

fields

variables
widgets and widget qualifiers
non-widget resource types (datasets, streams...)
system handles
SQL
object-oriented usage
filenames, symbol use cases (these may be tricky)

> I have checked once again all already marked categories in gaps/gap_analysis_marking.xml and have not found something is already marked.

I think you have missed some. For example, COM_PROPERTY and COM_METHOD are missing.

> The exception is probably KW_BROWSE as properly marked in gaps/lang_stmts.rule file with map addLangStatements and should be removed from the list.

No, I think you are confused here.

KW_BROWSE is used in multiple places. The DEFINE BROWSE case cannot be matched from expr. The one that can be matched from expr is in lvalue. This is completely different. It will be used for something like BROWSE my-browse:some-attribute or for a widget_qualifier like some-widget:some-attribute IN BROWSE my-browse.

#152 - 03/28/2017 08:53 PM - Eugenie Lyzenko

New keywords to process:

```
COM_METHOD
COM_PROPERTY
KW_IN
KW_SELECT
KW_ESCAPE
COLON
KW_AS
KW_BY_PTR
KW_BY_VAR_P
STREAM
```

record_phrase:

```
BUFFER
```

```
TABLE
TEMP_TABLE
WORK_TABLE
```

Class based property access:

```
ATTR_CLASS
```

FUNC_COM_HANDLE already marked as other FUNC_* tokens, right?

Also there are not yet marked literals:

```
KW_FIND_NO
KW_FIND_PO
KW_FIND_CS
KW_FIND_GLO
KW_FIND_WA
KW_ROW_UMOD
KW_ROW_DELD
KW_ROW_MODD
KW_ROW_CRTD
KW_WIN_DMIN
KW_WIN_MAX
KW_WIN_MIN
KW_WIN_NORM
```

According to Progress official docs the expression consist of:

Any one of the following ABL elements that represents or returns a value with a data type that is compatible with the expression data type, including:

- A literal (constant) value represented according to its data type (Data types)
- A database or temp-table field reference (Record phrase, DEFINE TEMP-TABLE statement)
- A reference to a variable scoped to the current procedure, user-defined function, or method of a class, or to an accessible class-based variable data member, including a subscripted or unsubscripted array reference (DEFINE VARIABLE statement, Class-based data member access)
- A reference to a parameter (of any mode) defined for the current procedure, user-defined function, or method of a class, including a subscripted or unsubscripted array reference (DEFINE PARAMETER statement)
- A reference to a readable class-based property or COM property, including a subscripted or unsubscripted array reference (DEFINE PROPERTY statement, Class-based property access, Accessing COM object properties and methods)
- Readable handle attribute reference (Accessing handle attributes and methods)
- Readable system handle reference (Handle Reference)
- An ABL built-in or user-defined function call (FUNCTION statement)
- A handle method, non-VOID COM method, or non-VOID class-based method call (Accessing handle attributes and methods, Accessing COM object properties and methods, Class-based method call)

Need to perform another progress.g scan to find out what can be missed more.

#153 - 03/29/2017 02:51 PM - Eugenie Lyzenko

This is the complete expr rules tree:

```
expr(KW_OR) ->
   log_and_expr(KW_AND) ->
      log_not_expr(KW_NOT) ->
         compare_expr(already marked) ->
            KW_LIKE(already marked as widget option)
            in_operator_parms ->
               LPARENS(already marked as operator)
               select_stmt ->
                  KW_SELECT(already marked as statement)
                  KW_ALL
                  KW_DISTINCT
                  sql_from ->
                     KW_FROM
                     sql_table_spec ->
                        record[true, false, false] ->
                           DB_SYMBOL
                           SYMBOL
            sub_select_stmt ->
               KW_ANY
               KW_ALL
               KW_SOME
               KW_EXISTS
               NOT_EXISTS
            sum_expr ->
               prod_expr ->
                  un_type ->
                     chained_object_members ->
                        primary_expr ->
                           sql_aggregate_funcs
                              KW_AVG
                              KW_COUNT
                              KW_MAX
                              KW_MIN
                              KW_SUM
                              MULTIPLY
                              KW_DISTINCT
                              any_non_reserved_symbol ->
                                 DB_SYMBOL (SYMBOL)
                                 FILENAME (SYMBOL)
                                 SYMBOL
                              seek_func
                                 FUNC_INT
                              cast_func
                                 FUNC_CLASS
                              sequence_funcs
                                 KW_CUR_VAL
                                 KW_NEXT_VAL
                              dynamic_function_func
                                 FUNC_POLY
                              frame_funcs
                                 FUNC_DEC
                                 FUNC_INT
                              record_funcs
                                 FUNC_*
                              accum_function
                                 FUNC_POLY
                              can_find_function
                                 FUNC_LOGICAL
                                 record_phrase ->
                                    record ->
                                       SYMBOL
                              super_function
                                 FUNC_POLY
                              postfix_funcs
                                 NOT_ENTERED
                              if_func
                                 FUNC_POLY
                           COM_INVOCATION
                           com_property_or_method ->
                              COM_METHOD
```

```
                        COM_PROPERTY
                        reserved_or_symbol ->
                           SYMBOL
                     downstream_chained_reference ->
                        method_call ->
                           METH_*
                        class_event ->
                           CLASS_EVENT
                           OBJECT_INVOCATION
                           any_non_reserved_symbol ->
                              DB_SYMBOL (SYMBOL)
                              FILENAME  (SYMBOL)
                        object_data_member ->
                           lvalue(line 27002 in progress.g) ->
                              WID_FRAME
                              QUERY
                              STREAM
                              VAR_*
                              WID_*
                              METH_*
                              FIELD_*
                              BUFFER, TABLE, TEMP_TABLE, WORK_TABLE
                              ...
                              widget_qualifier ->
                                 KW_IN
                        any_symbol_at_all ->
                           BOOL_TRUE (SYMBOL)
                           BOOL_FALSE (SYMBOL)
                           DB_SYMBOL (SYMBOL)
                           FILENAME  (SYMBOL)
           and_expr(KW_AND) ->
              sum_expr ->
                 See chain above ...
           escape_char ->
              KW_ESCAPE
              STRING
           in_operator_sub_select_stmt ->
              NOT_IN
              KW_IN
              in_operator_parms ->
                 See chain above ...
```

Looking for possible missed keywords.

#154 - 03/29/2017 03:51 PM - Eugenie Lyzenko

The candidate list in note #149 has been updated based on recent research. Start separating the keywords for standalone groups noted in #151.

#155 - 03/29/2017 10:15 PM - Eugenie Lyzenko

Here is the expressions related keywords distributed to the respective groups. We can create either single file for all of them or put every group in standalone file. I offer the second option - to use one file per group to simplify control. Is it OK?

fields
FIELD_BLOB
FIELD_CHAR
FIELD_CLASS
FIELD_CLOB
FIELD_COM_HANDLE
FIELD_DATE
FIELD_DATETIME
FIELD_DATETIME_TZ
FIELD_DEC
FIELD_HANDLE
FIELD_INT
FIELD_INT64
FIELD_LOGICAL
FIELD_RAW
FIELD_RECID
FIELD_ROWID

variables
KW_BETWEEN
KW_BY_PTR
KW_BY_VAR_P
KW_IS
VAR_BYTE
VAR_CHAR
VAR_CLASS
VAR_COM_HANDLE
VAR_DATE
VAR_DATETIME
VAR_DATETIME_TZ
VAR_DEC
VAR_DOUBLE
VAR_FLOAT
VAR_HANDLE
VAR_INT
VAR_INT64
VAR_LOGICAL
VAR_LONG
VAR_LONGCHAR
VAR_MEMPTR
VAR_RAW
VAR_RECID
VAR_ROWID
VAR_SHORT
VAR_USHORT

widgets and widget qualifiers
KW_AS
KW_BROWSE
KW_FIND_CS
KW_FIND_GLO
KW_FIND_NO
KW_FIND_PO
KW_FIND_WA
KW_WIN_DMIN
KW_WIN_MAX
KW_WIN_MIN
KW_WIN_NORM
METH_CHAR
METH_CLASS
METH_COM_HANDLE
METH_DATE
METH_DATETIME
METH_DATETIME_TZ
METH_DEC
METH_HANDLE

METH_INT
METH_INT64
METH_LOGICAL
METH_LONGCHAR
METH_MEMPTR
METH_POLY
METH_RAW
METH_RECID
METH_ROWID
METH_VOID
WID_BROWSE
WID_BUTTON
WID_COMBO
WID_DIALOG
WID_EDITOR
WID_FILL_IN
WID_FRAME
WID_IMAGE
WID_LITERAL
WID_MENU
WID_MENU_ITM
WID_RADIO
WID_RECT
WID_SEL_LST
WID_SLIDER
WID_SUB_MENU
WID_TEXT
WID_TOGGLE
WID_WINDOW

non-widget resource types (datasets, streams...)
BUFFER
KW_DATA_REL
KW_DISTINCT
KW_DSET_HND
KW_STRM_HND
TABLE
TEMP_TABLE
WORK_TABLE

system handles
SYS_HANDLE

SQL
KW_ALL
KW_ANY
KW_EXISTS
KW_FROM
KW_IN
KW_ROW_CRTD
KW_ROW_DELD
KW_ROW_MODD
KW_ROW_UMOD
KW_SELECT
KW_SOME
NOT_BETWEEN
NOT_EXISTS
NOT_IN
NOT_LIKE
QUERY

object-oriented usage
IS_NOT_NULL
IS_NULL
ATTR_CLASS
CLASS_EVENT
CLASS_NAME
COLON
COM_INVOCATION
COM_METHOD
COM_PROPERTY
OBJECT_INVOCATION

filenames, symbol use cases (these may be tricky)
BOOL_FALSE (SYMBOL, marked as known literal)

BOOL_TRUE (SYMBOL, marked as known literal)
DB_SYMBOL (SYMBOL)
FILENAME (SYMBOL)
KW_ESCAPE
STREAM
STRING
SYMBOL
UNKNOWN_VAL (SYMBOL, marked as known literal)

#156 - 03/30/2017 03:41 AM - Greg Shah

KW_BETWEEN
KW_BY_PTR
KW_BY_VAR_P
KW_IS

These are not variables. All variables are VAR_ only. Please look more carefully at where they are matched in the parser.

KW_BETWEEN is SQL. KW_IS is used in the SQL section but it is dropped and never appears in the tree. The result appears as IS_NOT_NULL or IS_NULL.

KW_BY_PTR and KW_BY_VAR_P are only used in COM support (which is its own category).

KW_FIND_CS
KW_FIND_GLO
KW_FIND_NO
KW_FIND_PO
KW_FIND_WA
KW_WIN_DMIN
KW_WIN_MAX
KW_WIN_MIN
KW_WIN_NORM

These are literals, they are not widgets.

METH_*

These are methods, which can sometimes be for OO and sometimes a handle based method. The handle based methods are already marked. The OO is not marked yet.

KW_ROW_CRTD
KW_ROW_DELD
KW_ROW_MODD
KW_ROW_UMOD

These are literals not SQL.

QUERY

Where is it used as SQL? I think this is just a resource.

IS_NOT_NULL
IS_NULL

SQL not OO

ATTR_CLASS

I think this can be treated as handle based attribute support, which we already mark. Although this is related to OO we will mark it differently.

COLON

This is handle-based method or attribute support. We don't mark it yet. It can be marked on its own.

COM_INVOCATION
COM_METHOD
COM_PROPERTY

No OO, this is COM support.

BOOL_FALSE (SYMBOL, marked as known literal)
BOOL_TRUE (SYMBOL, marked as known literal)
DB_SYMBOL (SYMBOL)
UNKNOWN_VAL (SYMBOL, marked as known literal)

These are all rewritten as SYMBOL, so they don't appear in the AST. They can be ignored.

FILENAME (SYMBOL)
SYMBOL

What cases are you referencing here?

KW_ESCAPE

SQL

STREAM

resource

STRING

Already marked as a literal. If it is used in a place where we rewrite the token type, then it won't appear as a STRING type so it can be ignored.

KW_IN

This is SQL. But it can also be seen in these places:

dynamic_function_func (which calls in_procedure_clause)
widget_qualifier

> We can create either single file for all of them or put every group in standalone file. I offer the second option - to use one file per group to simplify control. Is it OK?

Keep these as new groups added to the expressions.rules. I don't want the extra files added, when this is all related to expression processing.

#157 - 03/30/2017 03:47 AM - Greg Shah

I think that primary_expr is missing from the expr call tree in #1521-153. Have you handled all of these cases?

#158 - 03/30/2017 03:47 AM - Greg Shah

Did you check to see if we are marking all of the "unusual" built-in functions properly? Here is a list:

sql_aggregate_funcs
seek_func
cast_func
sequence_funcs
dynamic_function_func
frame_funcs
record_funcs
accum_function
can_find_function
super_function
postfix_funcs
if_func

If these are all properly parsed as FUNC_ with a "builtin" annotation, then they are already handled. Please confirm this.

#159 - 03/30/2017 03:48 AM - Greg Shah

What about DB_REF_NON_STATIC?

#160 - 03/30/2017 07:40 AM - Eugenie Lyzenko

Greg Shah wrote:

> What about DB_REF_NON_STATIC?

I thought this was replaced with SYMBOL:

```
DB_REF_NON_STATIC^ sym1:reserved_or_symbol { #sym1.setType(SYMBOL); }
```

#161 - 03/30/2017 08:21 AM - Greg Shah

> I thought this was replaced with SYMBOL:
>
> ```
> DB_REF_NON_STATIC^ sym1:reserved_or_symbol { #sym1.setType(SYMBOL); }
> >
> ```

No, it is not. It is the sym1 that is set to SYMBOL. sym1 is the returned AST node from reserved_or_symbol.

DB_REF_NON-STATIC will be the root node of the tree that results and it will have a first child that is a SYMBOL.

#162 - 03/30/2017 12:46 PM - Eugenie Lyzenko

I have added primary_expr with these sub-rules into general expr schema. As to "unusual" built-in functions, almost all are related to FUNC_* keywords. The newly found keywords are already marked as built-in functions and operators:

```
KW_AVG
KW_COUNT
KW_MAX
KW_MIN
KW_SUM
MULTIPLY
```

So they do not need to be marked as part of the expressions, correct?

#163 - 03/30/2017 01:21 PM - Greg Shah

KW_AVG
KW_COUNT
KW_MAX
KW_MIN
KW_SUM

These are already handled.

MULTIPLY

This is handled today for the prod_expr scenario (operators).

BUT, this is NOT handled today when it comes from sql_aggregate_funcs. This use case is SQL and is different. It still needs marking.

#164 - 03/30/2017 03:08 PM - Eugenie Lyzenko

Greg Shah wrote:

KW_FIND_CS
KW_FIND_GLO
KW_FIND_NO
KW_FIND_PO
KW_FIND_WA
KW_WIN_DMIN
KW_WIN_MAX
KW_WIN_MIN
KW_WIN_NORM

These are literals, they are not widgets.

OK. But what to do with them? Add to existed marking for literals? Or we need to create literal related group for not yet marked ones?

QUERY

Where is it used as SQL? I think this is just a resource.

This is from object_data_member -> lvalue as resource reference or browse widget related. Anyway you are right, this is not SQL.

> FILENAME (SYMBOL)
> SYMBOL

> What cases are you referencing here?

any_non_reserved_symbol. It is just a SYMBOL

#165 - 03/30/2017 03:12 PM - Greg Shah

> Add to existed marking fro literals?

Yes, just add them to the map of literals. Why do anything else?

> It is just a SYMBOL

Then FILENAME does not need marking directly, just put rules in to match the different SYMBOL cases that are possible in expressions.

#166 - 03/30/2017 03:14 PM - Eugenie Lyzenko

> Add to existed marking fro literals?

> Yes, just add them to the map of literals. Why do anything else?

OK.

#167 - 03/30/2017 03:23 PM - Eugenie Lyzenko

Updated and fixed keywords distribution into the groups:

fields
FIELD_BLOB
FIELD_CHAR
FIELD_CLASS
FIELD_CLOB
FIELD_COM_HANDLE
FIELD_DATE
FIELD_DATETIME
FIELD_DATETIME_TZ
FIELD_DEC
FIELD_HANDLE
FIELD_INT
FIELD_INT64
FIELD_LOGICAL
FIELD_RAW
FIELD_RECID
FIELD_ROWID

variables
VAR_BYTE
VAR_CHAR
VAR_CLASS
VAR_COM_HANDLE
VAR_DATE
VAR_DATETIME
VAR_DATETIME_TZ
VAR_DEC
VAR_DOUBLE
VAR_FLOAT
VAR_HANDLE
VAR_INT
VAR_INT64
VAR_LOGICAL
VAR_LONG
VAR_LONGCHAR
VAR_MEMPTR
VAR_RAW
VAR_RECID
VAR_ROWID
VAR_SHORT
VAR_USHORT

widgets and widget qualifiers
KW_AS
KW_BROWSE
WID_BROWSE
WID_BUTTON
WID_COMBO
WID_DIALOG
WID_EDITOR
WID_FILL_IN
WID_FRAME
WID_IMAGE
WID_LITERAL
WID_MENU
WID_MENU_ITM
WID_RADIO
WID_RECT
WID_SEL_LST
WID_SLIDER
WID_SUB_MENU
WID_TEXT
WID_TOGGLE
WID_WINDOW

non-widget resource types (datasets, streams...)
BUFFER
KW_DATA_REL
KW_DISTINCT

KW_DSET_HND
KW_STRM_HND
QUERY
STREAM
TABLE
TEMP_TABLE
WORK_TABLE

system handles
SYS_HANDLE

SQL
IS_NOT_NULL
IS_NULL
KW_ALL
KW_ANY
KW_BETWEEN
KW_ESCAPE
KW_EXISTS
KW_FROM
KW_IN
KW_SELECT
KW_SOME
MULTIPLY
NOT_BETWEEN
NOT_EXISTS
NOT_IN
NOT_LIKE

object-oriented usage
CLASS_EVENT
CLASS_NAME
METH_CHAR
METH_CLASS
METH_COM_HANDLE
METH_DATE
METH_DATETIME
METH_DATETIME_TZ
METH_DEC
METH_HANDLE
METH_INT
METH_INT64
METH_LOGICAL
METH_LONGCHAR
METH_MEMPTR
METH_POLY
METH_RAW
METH_RECID
METH_ROWID
METH_VOID
OBJECT_INVOCATION

filenames, symbol use cases (these may be tricky)
SYMBOL

COM support
COM_INVOCATION
COM_METHOD
COM_PROPERTY
KW_BY_PTR
KW_BY_VAR_P

handle-based methods or attributes
COLON

dynamic tables or field references
DB_REF_NON_STATIC

#168 - 03/30/2017 03:43 PM - Greg Shah

Ignore FILENAME, in expressions it needs no marking.

COLON needs to be handled. It should be handled on its own. It is used for handle-based methods and attributes, it is fully supported.

DB_REF_NON_STATIC is not a resource. It is a special syntax for dynamic table or field references. Mark it on its own. I believe it is fully supported.

#169 - 03/30/2017 03:52 PM - Eugenie Lyzenko

Greg Shah wrote:

> Ignore FILENAME, in expressions it needs no marking.
>
> COLON needs to be handled. It should be handled on its own. It is used for handle-based methods and attributes, it is fully supported.
>
> DB_REF_NON_STATIC is not a resource. It is a special syntax for dynamic table or field references. Mark it on its own. I believe it is fully supported.

OK. I have updated the fixed keywords list in #1521-167 above.

#170 - 03/30/2017 07:25 PM - Eugenie Lyzenko

Greg Shah wrote:

> > Add to existed marking for literals?
>
> Yes, just add them to the map of literals. Why do anything else?

Question for mentioned literals. These are known as "compiler constants" and having predefined values(0,1,2,...). This means all of them are substituted with real values on conversion stage and we have full support for them. Correct?

#171 - 03/31/2017 03:21 AM - Greg Shah

> These are known as "compiler constants" and having predefined values(0,1,2,...).

Yes.

> This means all of them are substituted with real values on conversion stage and we have full support for them. Correct?

No. Look at convert/literals.rules.

#172 - 03/31/2017 08:34 PM - Eugenie Lyzenko

Thinking about the way to find out the support status info for expression related keywords. Looks like the simple search in conversion rules for keyword value does not provide clear data. So the background to describe the support level is detailing investigation of the convert/expressions.rules, correct? Because for example complete button widget support does not mean the button reference support for usage within expression. Is it correct understanding?

#173 - 04/01/2017 12:58 PM - Greg Shah

> So the background to describe the support level is detailing investigation of the convert/expressions.rules, correct?

No. That ruleset is only designed to put in processing that occurs when the EXPRESSION node is being visited in the tree walk. It doesn't tell you about all the other usage.

There are many other rulesets involved (at least these from convert/, maybe others)

accumulate.rules
assignments.rules
builtin_functions.rules
control_flow.rules
database_access.rules
database_references.rules
expressions.rules
literals.rules
methods_attributes.rules
operators.rules
user_functions.rules
variable_references.rules
widget_references.rules

#174 - 04/05/2017 06:32 PM - Eugenie Lyzenko

Greg,

Support level marking question:
Assume for the button widget all references are supported within expression except button:get-dropped-file(). Does it mean the button's entry should be:

```
<rule>widAndQual.put(prog.wid_button  , rw.cvt_lvl_partial    | rw.rt_lvl_partial)</rule>
```

Or constant *_lvl_full_restr is more correct describing the support level?

#175 - 04/05/2017 10:02 PM - Eugenie Lyzenko

Searched widget support level. And found all of them except wid_sub_menu has partial support level. This means there are attributes or methods that are not supported but can potentially be referenced via widget object. The example for method is widget:get-dropped-file(), the example of the attribute is widget:html-charset.

#176 - 04/06/2017 05:36 AM - Greg Shah

Eugenie Lyzenko wrote:

> Searched widget support level. And found all of them except wid_sub_menu has partial support level. This means there are attributes or methods that are not supported but can potentially be referenced via widget object. The example for method is widget:get-dropped-file(), the example of the attribute is widget:html-charset.

No, we can mark these as rw.cvt_lvl_full | rw.rt_lvl_full. You are marking the widget not the attribute or method. We have separate marking for the attributes and methods, so we will find those gaps already. We also don't have to worry about the widget options because these are separately marked.

The questions for the widgets:

1. Do we support the inclusion of the widget type in a frame and in a CREATE <WIDGET > statement? If both are supported then the conversion level is full.

2. Do we have an implementation of the widget that includes both server and client code that is more than a "stub"? If so, then the widget can really be used at runtime and the support is full.

I believe these are the only restrictions:

slider widgets have no ChUI runtime support so it is partial runtime
control frame is support level none and none

#177 - 04/06/2017 12:58 PM - Eugenie Lyzenko

Rebased task branch 1521c from P2J trunk revision 11146. New revision is 11147.

#178 - 04/06/2017 04:22 PM - Eugenie Lyzenko

Greg Shah wrote:

> Eugenie Lyzenko wrote:
> The questions for the widgets:
>
> 1. Do we support the inclusion of the widget type in a frame and in a CREATE <WIDGET > statement? If both are supported then the conversion level is full.

Yes, we have the support.

> 2. Do we have an implementation of the widget that includes both server and client code that is more than a "stub"? If so, then the widget can really be used at runtime and the support is full.
>
> I believe these are the only restrictions:
>
> slider widgets have no ChUI runtime support so it is partial runtime
> control frame is support level none and none

We do not even have the WID_* keyword for control frame widget. Is it OK at this time for progress.g?

#179 - 04/07/2017 08:33 PM - Eugenie Lyzenko

Task branch 1521c for review updated to revision 11148.

This is not yet complete version. Items under investigation is commented out. Not clear for following keywords:

```
kw_data_rel
kw_distinct
kw_dset_hnd
```

I declared all of them as having full support. Although I have not found any references in riles file but in progress.g the keywords are taking into account and transforming into other language elements and properly converted. Is it correct approach? There are other keywords with similar handling so I need to know if I'm on right way.

#180 - 04/08/2017 09:22 AM - Greg Shah

> I declared all of them as having full support.

No, it is incorrect. All of these are NONE.

> Although I have not found any references in riles file but in progress.g the keywords are taking into account and transforming into other language elements

I think you are mistaken. What makes you think they are transformed into other elements?

#181 - 04/08/2017 11:28 AM - Eugenie Lyzenko

Greg Shah wrote:

> I think you are mistaken. What makes you think they are transformed into other elements?

For example kw_data_rel:

```
data_relation! [int idx]
   :
      dr:KW_DATA_REL (d:symbol)? f:for_data_relation_spec
      {
         if (#d == null)
         {
            // create an artificial node
            #d = #[SYMBOL, String.format("Relation%d", idx)];
         }

         String name = #d.getText();

         // add to the relation namespace
         sym.addDataRelation(name, DATA_RELATION);
...
```

then

```
...
            case KW_DATA_REL:
               // not reserved, we have to look it up
               qualType = sym.lookupDataRelation(LT(2).getText());
               break;
...
         options { generateAmbigWarnings = false; }
         :
           // widget, buffer, temp-table or query qualifier path
```

```
            { qualifier }?
            (
                 (
                      KW_QUERY^
                    | KW_BROWSE^
                    | KW_MENU^
                    | KW_SUB_MENU^
                    | KW_MENU_ITM^
                    | KW_DATASET^
                    | KW_DATA_SRC^
                    | KW_DATA_REL^
                 )
                 w:symbol                          { #w.setType(qualType); }
...
```

#182 - 04/08/2017 11:46 AM - Greg Shah

You are mis-interpreting the parser.

This code:

```
            // create an artificial node
            #d = #[SYMBOL, String.format("Relation%d", idx)];
```

will create nodes in the tree.

Likewise the qualifier usage will appear in the tree for references to the data relation. These are just not yet supported in the downstream conversion or runtime.

#183 - 04/08/2017 11:56 AM - Greg Shah

> We do not even have the WID_* keyword for control frame widget. Is it OK at this time for progress.g?

Yes, this is OK. We only use WID_ constants for static widgets (widgets that can be implicit or explicitly defined or referenced by name). A control frame can only be created dynamically (using a CREATE CONTROL-FRAME) and thus only referenced by a handle instance. There is no need for a WID_CONTROL_FRAME type.

#184 - 04/10/2017 03:18 PM - Eugenie Lyzenko

Task branch 1521c for review updated to revision 11149.

This is the next step update. For OO and SQL groups. The support status is defined based on rule files current implementation. However I'm not 100% sure everything is correct. Because the support criteria is not completely clear for me. If the keyword is not encountered in rule file - does it always mean the support level is none?

The remaining keyword is SYMBOL. As I recall there were notes for this keyword, need to perform special processing here. Do you have any notes for gaps/expressions.rules content of this keyword?

#185 - 04/11/2017 05:25 AM - Greg Shah

> If the keyword is not encountered in rule file - does it always mean the support level is none?

Usually, yes.

If there are no conversion rules that reference that token, then how can the code show up in the converted system?

#186 - 04/11/2017 05:26 AM - Greg Shah

> The remaining keyword is SYMBOL. As I recall there were notes for this keyword, need to perform special processing here. Do you have any notes for gaps/expressions.rules content of this keyword?

It will depend on the list of use cases. Make the list here and I will share specific thoughts.

#187 - 04/11/2017 08:27 AM - Eugenie Lyzenko

Greg Shah wrote:

> It will depend on the list of use cases. Make the list here and I will share specific thoughts.

SYMBOL related keywords. From progress.g:

```
any_non_reserved_symbol
   DB_SYMBOL
```

```
   FILENAME
```

But as we discussed previously in addition to:

```
BOOL_FALSE
BOOL_TRUE
```

all of them are rewritten by SYMBOL keyword and can be ignored. So we can just mark as having full support, right?

Also everything covered by SYMBOL case in progress.g(line 31496) including:

```
DIVIDE
BACKSLASH
LETTER
COLON
DB_REF_NON_STATIC
COMMA
EQUALS
NOT_EQ
GT
LT
GTE
LTE
LPARENS
RPARENS
LBRACKET
RBRACKET
MULTIPLY
AT
CARET
NUM_LITERAL
UNKNOWN_TOKEN
```

can be ignored as well?

#188 - 04/17/2017 01:46 PM - Eugenie Lyzenko

Task branch 1521c for review updated to revision 11150.

All changes for only gaps/expressions.rule. This update fixes the rule line issue with missing </rule> brace at the end. Also fixed the map names Camel Case issue. The new support status has been added to the keywords: KW_AS and SYMBOL, full level for both in conversion and runtime.

#189 - 04/17/2017 06:05 PM - Eugenie Lyzenko

Rebased task branch 1521c from P2J trunk revision 11147. New revision is 11151.

#190 - 04/18/2017 06:26 PM - Eugenie Lyzenko

Rebased task branch 1521c from P2J trunk revision 11148. New revision is 11152.

#191 - 04/21/2017 02:29 PM - Eugenie Lyzenko

Greg,

While I'm working on another task should I keep rebasing 1521c branch with the upcoming new trunk versions?

#192 - 04/21/2017 02:52 PM - Greg Shah

No, I'll handle it.

#193 - 04/21/2017 02:53 PM - Eugenie Lyzenko

Greg Shah wrote:

> No, I'll handle it.

OK.

#194 - 06/15/2017 02:24 PM - Greg Shah

On March 24, 2017 we merged branch 1521b into the trunk as revision 11144. Branch 1521b has been archived.

#195 - 03/09/2018 09:59 AM - Greg Shah

1521c was merged into 1514a which was merged to trunk as rev 11157 on 2017-08-08. 1521c was archived at that time.

#196 - 11/09/2019 08:50 AM - Greg Shah

We need to add some additional support levels:

- cvt_lvl_no_need/rt_lvl_no_need
  - Description: "The feature cannot and will not be implemented because it is unnecessary and/or has no meaning/purpose in the Java implementation. This most often occurs because a feature is implementation specific to the 4GL AND has no application visible behavior to be implemented."
  - In reports it should be colored some different but dark shade of green since there is nothing to do there.
- cvt_lvl_no_plan/rt_lvl_no_plan
  - Description: "An implementation is possible but there is no plan to implement this feature at this time. This most often occurs because a feature has no common business logic purpose and/or is only used for utilities/admin/support instead of end user code."
  - In reports it should be colored some shade of orange. Although it is not something to be implemented in FWD, customers need to do something with the 4GL code to drop or bypass this usage. This means it is not a green item.

#197 - 11/13/2019 12:09 PM - Greg Shah

- Assignee deleted (Eugenie Lyzenko)