Base Language - Bug #4476

Shared frame does not get @ widget method generated

12/13/2019 03:40 PM - Roger Borrello

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

History

#1 - 12/13/2019 03:58 PM - Roger Borrello

Testcase

20200102 Update: rules/fixups/inline_format_phrase_var_defs.rules checked into **4207a-11369**.

Added 2 testcases (since it is a shared frame)

uast/sharedframes/shared_frame_at_label-run.p

```
def new shared frame sf1.
def new shared var fieldx as char no-undo.

form fieldx
  label "Label" at 2 myat no-label as char
  with frame sf1.

run z:\uast\sharedframes\shared_frame_at_label.p.
message "Done.".
```

uast/sharedframes/shared_frame_at_label.p

```
def shared frame sf1.
def shared var fieldx as char no-undo.

form fieldx
  label "Label" at 2 myat no-label as char
  with frame sf1.

display "Locale" @ myat with frame sf1.

message "Done.".
```

Errors

Conversion Error(s)

Maybe not errors, but could help.

```
     [java] ------------------------------------------------------------------------------
     [java] Detecting Frame Interfaces
     [java] ------------------------------------------------------------------------------
     [java]
     [java] ./abl/shared_frame_at_label.p
     [java] WARNING: Found 1 orphan SHARED FRAME(S):
     [java]     FrameSf1_1
     [java] Elapsed job time:  00:00:00.017
     [java] 
     [java] ------------------------------------------------------------------------------
     [java] Frame Generator
     [java] ------------------------------------------------------------------------------
     [java] 
     [java] ./abl/shared_frame_at_label.p
     [java] INFO: Created dummy interface for 'FrameSf1_1' orphan SHARED FRAME
     [java] Elapsed job time:  00:00:00.282
     [java] 
```

Javac Error(s)

```
    [javac] projects/testcases/uast/uast/buildarea/src/com/goldencode/testcases/abl/SharedFrameAtLabel.java:32
: error: cannot find symbol
    [javac]             new Element("Locale", "x(6)", sf1Frame.widgetMyat())
    [javac]                                                  ^
    [javac]   symbol:   method widgetMyat()
    [javac]   location: variable sf1Frame of type FrameSf1_1
    [javac] 1 error
```

Generated Java Files

In ui/shared_frames/FrameSf1_1.java:

```
package com.goldencode.testcases.ui.shared_frames;

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

public interface FrameSf1_1
extends CommonFrame
{
   public character getFieldx();

   public void setFieldx(character parm);

   public void setFieldx(String parm);

   public void setFieldx(BaseDataType parm);

   public FillInWidget widgetFieldx();

}
```

In abl/SharedFrameAtLabel.java:

```
package com.goldencode.testcases.abl;

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

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

/**
 * Business logic (converted to Java from the 4GL source code
 * in abl/shared_frame_at_label.p).
 */
public class SharedFrameAtLabel
{
   FrameSf1_1 sf1Frame = GenericFrame.importSharedFrame(FrameSf1_1.class, "sf1");

   character fieldx = SharedVariableManager.lookupVariable("fieldx", character.class, true);

   /**
    * External procedure (converted to Java from the 4GL source code
    * in abl/shared_frame_at_label.p).
    */
   public void execute()
   {
      character myat = UndoableFactory.character();

      externalProcedure(SharedFrameAtLabel.this, new Block((Body) () -> 
      {
         FrameElement[] elementList0 = new FrameElement[]
         {
            new Element("Locale", "x(6)", sf1Frame.widgetMyat())
         };

         sf1Frame.display(elementList0);

         message("Done.");
      }));
   }
}
```

#3 - 12/16/2019 09:26 AM - Roger Borrello

Roger Borrello wrote:

> ## Testcase
>
> Added 2 testcases (since it is a shared frame)
>
> [...]
>
> [...]
>
> ## Errors
>
> ### Conversion Error(s)
>
> Maybe not errors, but could help.
>
> [...]

Javac Error(s)

[...]

Generated Java Files

In ui/shared_frames/FrameSf1_1.java:

[...]

In abl/SharedFrameAtLabel.java:

[...]

There are other cases that have failed, and are all very similar, related to **shared frames** which are used by forms that contain declarations of widgets *on-the-fly* (for lack of a better term).

Another similar case...

```
def shared frame sf1.
def shared var fieldx as char no-undo.

form
  mychar as char format "x(6)" no-label space
  with frame sf1.

display "Locale" mychar with frame sf1.

message "Done.".
```

#4 - 12/31/2019 02:14 PM - Roger Borrello

Let me recap what is generated by the pair of testcases in #note-1 :

sharedframes/SharedFrameAtLabel.java

```
package com.goldencode.testcases.sharedframes;

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

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

/**
 * Business logic (converted to Java from the 4GL source code
 * in sharedframes/shared_frame_at_label.p).
 */
public class SharedFrameAtLabel
{
   FrameSf1_1 sf1Frame = GenericFrame.importSharedFrame(FrameSf1_1.class, "sf1");

   character fieldx = SharedVariableManager.lookupVariable("fieldx", character.class, true);

   /**
    * External procedure (converted to Java from the 4GL source code
    * in sharedframes/shared_frame_at_label.p).
    */
   public void execute()
   {
      character myat = UndoableFactory.character();

      externalProcedure(SharedFrameAtLabel.this, new Block((Body) () -> 
      {
         FrameElement[] elementList0 = new FrameElement[]
         {
            new Element("Locale", "x(6)", sf1Frame.widgetMyat())
         };

         sf1Frame.display(elementList0);

         message("Done.");
      }));
   }
}
```

ui/shared_frames/FrameSf1.java

```
package com.goldencode.testcases.ui.shared_frames;

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

public interface FrameSf1_1
extends CommonFrame
{
   public character getFieldx();

   public void setFieldx(character parm);

   public void setFieldx(String parm);

   public void setFieldx(BaseDataType parm);

   public FillInWidget widgetFieldx();

}
```

sharedframes/SharedFrameAtLabelRun.java

```
package com.goldencode.testcases.sharedframes;
```

```
import com.goldencode.p2j.util.*;
import com.goldencode.testcases.ui.sharedframes.*;
import com.goldencode.p2j.ui.*;

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

/**
 * Business logic (converted to Java from the 4GL source code
 * in sharedframes/shared_frame_at_label-run.p).
 */
public class SharedFrameAtLabelRun
{
   SharedFrameAtLabelRunSf1 sf1Frame = GenericFrame.createSharedFrame(SharedFrameAtLabelRunSf1.class, "sf1");

   character fieldx = SharedVariableManager.addVariable("fieldx", true, new character(""));

   /**
    * External procedure (converted to Java from the 4GL source code
    * in sharedframes/shared_frame_at_label-run.p).
    */
   public void execute()
   {
      character myat = UndoableFactory.character();

      externalProcedure(SharedFrameAtLabelRun.this, new Block((Body) () ->
      {
         sf1Frame.openScope();
         ControlFlowOps.invoke("/testcases/uast/sharedframes/shared_frame_at_label.p");
         message("Done.");
      }));
   }
}
```

ui/sharedframes/SharedFrameAtLabelRunSf1.java

```
package com.goldencode.testcases.ui.sharedframes;

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

public interface SharedFrameAtLabelRunSf1
extends FrameSf1_1
{
   public static final Class configClass = SharedFrameAtLabelRunSf1Def.class;

   public character getFieldx();

   public void setFieldx(character parm);

   public void setFieldx(String parm);

   public void setFieldx(BaseDataType parm);

   public FillInWidget widgetFieldx();

   public character getMyat();

   public void setMyat(character parm);

   public void setMyat(String parm);

   public void setMyat(BaseDataType parm);

   public FillInWidget widgetMyat();

   public static class SharedFrameAtLabelRunSf1Def
   extends WidgetList
   {
      FillInWidget fieldx = new FillInWidget();
      FillInWidget myat = new FillInWidget();

      public void setup(CommonFrame frame)
      {
         frame.setDown(1);
         fieldx.setDataType("character");
         fieldx.setForceLabel(true);
         fieldx.setLabel("Label");
         fieldx.setColumn(2);
         myat.setDataType("character");
         myat.setNoLabels(true);
         fieldx.setFormat("x(8)");
         myat.setLabel("myat");
         myat.setFormat("x(8)");
      }

      {
         addWidget("fieldx", "fieldx", fieldx);
         addWidget("myat", "myat", myat);
      }
   }
}
```

#5 - 12/31/2019 03:04 PM - Roger Borrello

This is data from a correctly built testcase, which does **not** create the fields in the from on-the-fly.

sharedframes/shared_frame_at-label2.p

```
def shared frame sf1.
def shared var fieldx as char no-undo.
def shared var myat as char.

form fieldx
  label "Label" at 2 myat no-label
  with frame sf1.

display "Locale" @ myat with frame sf1.

message "Done.".
```

sharedframes/SharedFrameAtLabel2.java

```
package com.goldencode.testcases.sharedframes;
```

```
import com.goldencode.p2j.util.*;
import com.goldencode.testcases.ui.shared_frames.*;
import com.goldencode.p2j.ui.*;

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

/**
 * Business logic (converted to Java from the 4GL source code
 * in sharedframes/shared_frame_at_label-2.p).
 */
public class SharedFrameAtLabel2
{
   FrameSf1_1 sf1Frame = GenericFrame.importSharedFrame(FrameSf1_1.class, "sf1");

   character fieldx = SharedVariableManager.lookupVariable("fieldx", character.class, true);

   character myat = SharedVariableManager.lookupVariable("myat", character.class);

   /**
    * External procedure (converted to Java from the 4GL source code
    * in sharedframes/shared_frame_at_label-2.p).
    */
   public void execute()
   {
      externalProcedure(SharedFrameAtLabel2.this, new Block((Body) () -> 
      {
         FrameElement[] elementList0 = new FrameElement[]
         {
            new Element("Locale", "x(6)", sf1Frame.widgetMyat())
         };

         sf1Frame.display(elementList0);

         message("Done.");
      }));
   }
}
```

ui/shared_frames/FrameSf1.java

```
package com.goldencode.testcases.ui.shared_frames;

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

public interface FrameSf1_1
extends CommonFrame
{
   public character getFieldx();

   public void setFieldx(character parm);

   public void setFieldx(String parm);

   public void setFieldx(BaseDataType parm);

   public FillInWidget widgetFieldx();

   public character getMyat();

   public void setMyat(character parm);

   public void setMyat(String parm);

   public void setMyat(BaseDataType parm);

   public FillInWidget widgetMyat();

}
```

sharedframes/shared_frame_at_label-run-2.p

```
def new shared frame sf1.
def new shared var fieldx as char no-undo.
def new shared var myat as char.

form fieldx
  label "Label" at 2 myat no-label
  with frame sf1.

run ~/testcases/uast/sharedframes/shared_frame_at_label.p.
message "Done.".
```

sharedframes/SharedFrameAtLabelRun2.java

```
package com.goldencode.testcases.sharedframes;

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

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

/**
 * Business logic (converted to Java from the 4GL source code
 * in sharedframes/shared_frame_at_label-run-2.p).
 */
public class SharedFrameAtLabelRun2
{
   SharedFrameAtLabelRun2Sf1 sf1Frame = GenericFrame.createSharedFrame(SharedFrameAtLabelRun2Sf1.class, "sf1")
;

   character fieldx = SharedVariableManager.addVariable("fieldx", true, new character(""));

   character myat = SharedVariableManager.addVariable("myat", new character(""));

   /**
    * External procedure (converted to Java from the 4GL source code
    * in sharedframes/shared_frame_at_label-run-2.p).
    */
   public void execute()
   {
      externalProcedure(SharedFrameAtLabelRun2.this, new Block((Body) () -> 
      {
         sf1Frame.openScope();
         ControlFlowOps.invoke("/testcases/uast/sharedframes/shared_frame_at_label.p");
         message("Done.");
      }));
   }
}
```

ui/sharedframes/SharedFrameAtLabelRun2Sf1.java

```
package com.goldencode.testcases.ui.sharedframes;

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

public interface SharedFrameAtLabelRun2Sf1
extends FrameSf1_1
{
   public static final Class configClass = SharedFrameAtLabelRun2Sf1Def.class;
   public character getFieldx();

   public void setFieldx(character parm);

   public void setFieldx(String parm);

   public void setFieldx(BaseDataType parm);

   public FillInWidget widgetFieldx();

   public character getMyat();

   public void setMyat(character parm);

   public void setMyat(String parm);

   public void setMyat(BaseDataType parm);

   public FillInWidget widgetMyat();

   public static class SharedFrameAtLabelRun2Sf1Def
   extends WidgetList
   {
      FillInWidget fieldx = new FillInWidget();

      FillInWidget myat = new FillInWidget();

      public void setup(CommonFrame frame)
      {
         frame.setDown(1);
         fieldx.setDataType("character");
         fieldx.setForceLabel(true);
         fieldx.setLabel("Label");
         fieldx.setColumn(2);
         myat.setDataType("character");
         myat.setNoLabels(true);
         fieldx.setFormat("x(8)");
         myat.setLabel("myat");
         myat.setFormat("x(8)");
      }

      {
         addWidget("fieldx", "fieldx", fieldx);
         addWidget("myat", "myat", myat);
      }
   }
}
```

#6 - 01/02/2020 10:02 AM - Roger Borrello

Comparing the AST generated for the broken case (shared_frame_at_label.p) against the correct case (shared_frame_at_label-2.p):

The working case has STATEMENT followed by KW_FORM:

```
    <ast col="0" id="12884901930" line="0" text="statement" type="STATEMENT">
      <ast col="1" id="12884901931" line="5" text="form" type="KW_FORM">
```

The broken case has STATEMENT with INLINE_VAR_DEF_ARRAY coming before the KW_FORM:

```
    <ast col="0" id="25769803804" line="0" text="statement" type="STATEMENT">
      <ast col="0" id="25769803866" line="0" text="" type="INLINE_VAR_DEF_ARRAY">
        <ast col="0" id="25769803867" line="0" text="format phrase" type="INLINE_VAR_DEF">
          <annotation datatype="java.lang.String" key="name" value="myat"/>
          <annotation datatype="java.lang.Long" key="type" value="373"/>
          <annotation datatype="java.lang.Boolean" key="vardef" value="true"/>
          <annotation datatype="java.lang.String" key="javaname" value="myat"/>
          <annotation datatype="java.lang.String" key="classname" value="character"/>
          <annotation datatype="java.lang.Long" key="defid" value="25769803823"/>
          <annotation datatype="java.lang.Long" key="peerid" value="197568495672"/>
          <ast col="0" id="25769803868" line="0" text="" type="BOGUS"/>
        </ast>
      </ast>
      <ast col="1" id="25769803805" line="4" text="form" type="KW_FORM">
```

#7 - 01/02/2020 10:07 AM - Greg Shah

That is probably not related. The issue will be the tree structure under the KW_FORM. The FORM_ITEM related to myat is not being matched properly in the shared frame interface calculations. Look in frame_generator.xml and search on "shared" (including the quotes). That annotation name is used to detect which frames are shared.

#8 - 01/02/2020 10:15 AM - Roger Borrello

Greg Shah wrote:

> That is probably not related. The issue will be the tree structure under the KW_FORM. The FORM_ITEM related to myat is not being matched properly in the shared frame interface calculations. Look in frame_generator.xml and search on "shared" (including the quotes). That annotation

> name is used to detect which frames are shared.

Well, I was about to post that the broken case has KW_AS appearing within the FORMAT_PHRASE:

```
          <ast col="0" id="25769803823" line="0" text="format phrase" type="FORMAT_PHRASE">
            <annotation datatype="java.lang.String" key="name" value="myat"/>
            <annotation datatype="java.lang.Long" key="type" value="373"/>
            <annotation datatype="java.lang.Boolean" key="vardef" value="true"/>
            <annotation datatype="java.lang.String" key="javaname" value="myat"/>
            <annotation datatype="java.lang.String" key="classname" value="character"/>
            <annotation datatype="java.lang.Long" key="frame-id" value="25769803865"/>
            <ast col="27" id="25769803824" line="5" text="no-label" type="KW_NO_LABEL">
              <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
            </ast>
            <ast col="36" id="25769803826" line="5" text="as" type="KW_AS">
              <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
              <ast col="39" id="25769803828" line="5" text="char" type="KW_CHAR"/>
            </ast>
          </ast>
```

Since the INLINE_VAR_DEF had type="BOGUS" I was guessing that the fact that there was a declaration correctly determined, but the type of the variable was missed, because it was further down the tree. I'll look at the matching of FORM_ITEM to myat but doesn't myat have to be defined properly, too?

#9 - 01/02/2020 10:23 AM - Greg Shah

> but doesn't myat have to be defined properly, too?

In the working case, the variable is not defined as an inline var def. You have an explicit DEF VAR myat.... That is why that case doesn't need the INLINE_VAR_DEF. Ignore it.

> I'll look at the matching of FORM_ITEM to myat

This is the key. This is probably a SYMBOL there instead of a VAR_CHAR. Code in frame_generator.xml is probably not deal with this case properly.

#10 - 01/02/2020 12:11 PM - Roger Borrello

Greg Shah wrote:

> > but doesn't myat have to be defined properly, too?
>
> In the working case, the variable is not defined as an inline var def. You have an explicit DEF VAR myat.... That is why that case doesn't need the INLINE_VAR_DEF. Ignore it.
>
> > I'll look at the matching of FORM_ITEM to myat
>
> This is the key. This is probably a SYMBOL there instead of a VAR_CHAR. Code in frame_generator.xml is probably not deal with this case properly.

Corrected code that was not looking for FORM_ITEM as a parent type in rules/fixups/inline_format_phrase_var_defs.rules. This rule adds a refid annotation to the symbol node and changes the token type as if this was a real variable reference. Since this is a kind of pre-reference to the var before the var as actually defined, the parser is not the easiest place to handle this.

rules/fixups/inline_format_phrase_var_defs.rules checked into **4207a-11369**.

#11 - 01/02/2020 12:12 PM - Roger Borrello

- Status changed from New to WIP

- % Done changed from 0 to 100

#12 - 01/02/2020 01:07 PM - Greg Shah

- Status changed from WIP to Test

#13 - 01/15/2020 11:02 AM - Roger Borrello

I see a difference in the generated frame definition between trunk, and this revision.

I modified the testcase at/at_frame_broken_display-standalone.p by adding a format "x(25)" before the @ base-ref:

```
def var mymess as character format "x(25)" no-undo init "mymess-v".
display "Missing value" format "x(25)" @ mymess with frame f1.
message "Done."
```

There are 2 methods no longer in the frame definition:

```
<           mymess.setAt(true);
<           mymess.setAtFormatLength(13);
```

#14 - 01/15/2020 01:31 PM - Roger Borrello

Roger Borrello wrote:

> I see a difference in the generated frame definition between trunk, and this revision.
>
> I modified the testcase at/at_frame_broken_display-standalone.p by adding a format "x(25)" before the @ base-ref:
>
> [...]
>
> There are 2 methods no longer in the frame definition:
> [...]

With trunk, there is not a difference between the frame definition code whether you have the format "x(25)" before the @ base-ref or not. In other words, we it should be there, regardless, due to the @ base-ref.

Just an aside, the business logic code defaults to the string length for the width when it is not specified:

```
<             new Element("Missing value", "x(25)", f1Frame.widgetMymess())
---
>             new Element("Missing value", "x(13)", f1Frame.widgetMymess())
```

#15 - 01/15/2020 03:40 PM - Roger Borrello

Roger Borrello wrote:

> Roger Borrello wrote:
>
>> I see a difference in the generated frame definition between trunk, and this revision.
>>
>> I modified the testcase at/at_frame_broken_display-standalone.p by adding a format "x(25)" before the @ base-ref:
>>
>> [...]
>>
>> There are 2 methods no longer in the frame definition:
>> [...]
>
> With trunk, there is not a difference between the frame definition code whether you have the format "x(25)" before the @ base-ref or not. In other words, we it should be there, regardless, due to the @ base-ref.
>
> Just an aside, the business logic code defaults to the string length for the width when it is not specified:
> [...]

Interesting difference in the AST. The VAR_CHAR for mymess in **trunk** has oldtype=2781 (NUM_LITERAL) and in **4207a** has oldtype=2782 (SYMBOL).

#16 - 01/15/2020 03:56 PM - Greg Shah

This is an expected consequence of the change. We are now creating the widget from the child of AT instead of at the expression node. I would have expected this to be VAR_CHAR but anyway it should not matter here.

#17 - 01/15/2020 04:13 PM - Roger Borrello

Greg Shah wrote:

> This is an expected consequence of the change. We are now creating the widget from the child of AT instead of at the expression node. I would have expected this to be VAR_CHAR but anyway it should not matter here.

I was looking at it, because now that we create the widget from AT instead of EXPRESSION we don't find AT as the next sibling (using **has_option**) in convert/frame_generator.xml where we trying to set setAt=true.

#18 - 01/15/2020 05:45 PM - Roger Borrello

Updated convert/frame_generator.xml in 4207a-11378

We need to look at our parent node for @ base-field, instead of in sibling relationship, since we moved the widget data.

#19 - 01/16/2020 09:10 AM - Roger Borrello

I just wanted to make a note that the 4GL behavior when including conflicting formatting between the variable definition and on the DISPLAY statement, the DISPLAY wins out (no matter how ridiculous!)

```
def var mymess as character format "x(25)" no-undo init "v".
display "Missing value" format "x(5)" @ mymess with frame f1.
```

Results in **"Missi"**

```
def var mymess as character format "x(2)" no-undo init "v".
display "Missing value" format "x(25)" @ mymess with frame f1.
```

Results in **"Missing value   "**

```
def var mymess as character format "x(25)" no-undo init "v".
display "Missing value" format "$9,999.99" @ mymess with frame f1.
```

Results in **"$M,iss.in"**

#20 - 01/16/2020 09:19 AM - Constantin Asofiei

Roger, please change mymess to be a decimal and use some similar Missing value - I wonder if @ option just formats and passes whatever is on its left to the target widget's screen-value, without doing any kind of validation that value and the target widget are compatible.

#21 - 01/16/2020 09:24 AM - Roger Borrello

Constantin Asofiei wrote:

> Roger, please change mymess to be a decimal and use some similar Missing value - I wonder if @ option just formats and passes whatever is on its left to the target widget's screen-value, without doing any kind of validation that value and the target widget are compatible.

Like so?

```
def var mymess as decimal format "99,999.99" no-undo init "1.00".
display "4444444" format "$9,999.99" @ mymess with frame f1.
```

Seven fours... Results in **"$4,444.44"**

#22 - 01/16/2020 09:25 AM - Constantin Asofiei

Roger Borrello wrote:

> Constantin Asofiei wrote:
>
>> Roger, please change mymess to be a decimal and use some similar Missing value - I wonder if @ option just formats and passes whatever is on its left to the target widget's screen-value, without doing any kind of validation that value and the target widget are compatible.
>
> Like so?
> [...]
> Seven fours... Results in **"$4,444.44"**

Yes, but use "Missing value" instead of 4444444.

#23 - 01/16/2020 09:27 AM - Roger Borrello

Constantin Asofiei wrote:

> Roger Borrello wrote:
>
>> Constantin Asofiei wrote:
>>
>>> Roger, please change mymess to be a decimal and use some similar Missing value - I wonder if @ option just formats and passes whatever is on its left to the target widget's screen-value, without doing any kind of validation that value and the target widget are compatible.
>>
>> Like so?
>> [...]
>> Seven fours... Results in **"$4,444.44"**
>
> Yes, but use "Missing value" instead of 4444444.

```
def var mymess as decimal format "99,999.99" no-undo init "1.00".
display "Missing value" format "$9,999.99" @ mymess with frame f1.
```

Results in **"$M,iss.in"**

#24 - 01/16/2020 09:33 AM - Greg Shah

Hmm, this is worse than what we already knew (which was really bad).

Questions:

- Does this happen for any @ usage (e.g. UPDATE or some other non-display case)?
- I remember thinking that one cannot edit such a field. But then again, I remember that the data type can be overridden in some way, and not just to character. So I wonder: can we find a way to allow the user to edit such a "corrupt" override?

#25 - 01/16/2020 09:36 AM - Roger Borrello

Greg Shah wrote:

> Hmm, this is worse than what we already knew (which was really bad).
>
> Questions:
>
> - Does this happen for any @ usage (e.g. UPDATE or some other non-display case)?
> - I remember thinking that one cannot edit such a field. But then again, I remember that the data type can be overridden in some way, and not just to character. So I wonder: can we find a way to allow the user to edit such a "corrupt" override?

Throw a few test samples in here and we can see.

#26 - 01/16/2020 09:46 AM - Greg Shah

> Throw a few test samples in here and we can see.

That is your task.

#27 - 01/17/2020 11:45 AM - Roger Borrello

Given the failed regression testing, I'm revisiting with this testcase:

```
def var mymess as decimal format "9.9" no-undo init "1.00".
display mymess with frame f1.
pause.
display "Missing value" format "x(10)" @ mymess with frame f1.
message "Done.".
```

The situation is the setAt() and setAtFormatLength(10) are not being emitted.

Debug was added with dumping the parent tree after we are to find the @ base field. Dump is whether found or not. In trunk, we have 2 places in the tree where the node is investigated... in 4207a, only one time.

The trunk execution shows:

```
     [java] ------------------------------------------------------------------------------
     [java] Frame Generator
     [java] ------------------------------------------------------------------------------
     [java] 
     [java] ./uast/at/at_base_field_max_quirky1.p
     [java] frame_generator (trunk): set_bool for setAt NOT called
     [java] display [KW_DISP]:12884901914 @2:1
     [java]    expression [EXPRESSION]:12884901916 @0:0 HIDDEN
     [java]       mymess [VAR_DEC]:12884901917 @2:9
     [java]    with [FRAME_PHRASE]:12884901919 @2:16
     [java]       frame [KW_FRAME]:12884901921 @2:21 HIDDEN
     [java]          f1 [WID_FRAME]:12884901923 @2:27
     [java] 
     [java] frame_generator (trunk): set_bool for setAt called
     [java] display [KW_DISP]:12884901931 @4:1
     [java]    expression [EXPRESSION]:12884901933 @0:0 HIDDEN
     [java]       "Missing value" [STRING]:12884901934 @4:9
     [java]    format phrase [FORMAT_PHRASE]:12884901936 @0:0
     [java]       format [KW_FORMAT]:12884901937 @4:25
     [java]          expression [EXPRESSION]:12884901939 @0:0
     [java]             "x(10)" [STRING]:12884901940 @4:32
     [java]       @ [AT]:12884901942 @4:40
     [java]          mymess [VAR_DEC]:12884901944 @4:42
     [java]    with [FRAME_PHRASE]:12884901946 @4:49
     [java]       frame [KW_FRAME]:12884901948 @4:54 HIDDEN
     [java]          f1 [WID_FRAME]:12884901950 @4:60
```

The 4207a execution shows:

```
     [java] ------------------------------------------------------------------------------
     [java] Frame Generator
     [java] ------------------------------------------------------------------------------
     [java] 
     [java] ./uast/at/at_base_field_max_quirky1.p
     [java] frame_generator: set_bool for setAt NOT called
     [java] display [KW_DISP]:12884901914 @2:1
     [java]    expression [EXPRESSION]:12884901916 @0:0 HIDDEN
     [java]       mymess [VAR_DEC]:12884901917 @2:9
     [java]    with [FRAME_PHRASE]:12884901919 @2:16
     [java]       frame [KW_FRAME]:12884901921 @2:21 HIDDEN
     [java]          f1 [WID_FRAME]:12884901923 @2:27
```

#28 - 01/17/2020 11:57 AM - Roger Borrello

We are only iterating through wlist once, because we no longer add the don't add @ as a widget, presumably when it was seen in annotations/frame_scoping.rules. It seems we lose some important formatting information in this case.

#29 - 01/17/2020 12:01 PM - Roger Borrello

Roger Borrello wrote:

> We are only iterating through wlist once, because we no longer add the don't add @ as a widget, presumably when it was seen in annotations/frame_scoping.rules. It seems we lose some important formatting information in this case.

Debug added to add_widget shows this is the case:

```
     [java] ------------------------------------------------------------------------------
     [java] Code Conversion Annotations
     [java] ------------------------------------------------------------------------------
     [java]
     [java] Optional rule set [customer_specific_annotations_prep] not found.
     [java] ./uast/at/at_base_field_max_quirky1.p
     [java] frame_scoping.add_widget: Not adding widget
     [java] format phrase [FORMAT_PHRASE]:12884901936 @0:0
     [java]    format [KW_FORMAT]:12884901937 @4:25
     [java]       expression [EXPRESSION]:12884901939 @0:0
     [java]          "x(10)" [STRING]:12884901940 @4:32
     [java]    @ [AT]:12884901942 @4:40
     [java]       mymess [VAR_DEC]:12884901944 @4:42
```

#30 - 01/19/2020 05:20 PM - Roger Borrello

I have a quiz for the experts... What data type should the getTmess() getter for the widget on frame1 return?

```
def var mymess as decimal no-undo init "1.00".
def var tmess as char.    /* Dummy var used in @ */
display int(mymess) format "-zzzzzz9"  @ tmess with frame f1.
message "Done.".
```

Hint... in *trunk*, the datatype changes if you remove the int() function.

#31 - 01/19/2020 05:30 PM - Greg Shah

I would expect the widget to have character type. But the override specified by the @ is going to be "temporarily" whatever the type is of the expression to the left of the @. We may be generating it incorrectly in trunk.

#32 - 01/19/2020 09:57 PM - Roger Borrello

Greg Shah wrote:

> I would expect the widget to have character type. But the override specified by the @ is going to be "temporarily" whatever the type is of the expression to the left of the @. We may be generating it incorrectly in trunk.

In trunk, it is integer (or decimal if you remove the int()). In **4207a** it is character, which I, too, believe is correct. It does leads to several diffs in the regression testing. The run-time situation is most likely not a problem, since the widget getter/setter aren't used when output to a file in a report. Not-to-mention setDataType(character) is emitted in both cases.

Now the formatting... that's being worked on, as we are missing that boat.

Trunk:

```
         frame.setDown(1);
         tmess.setAt(true);
         tmess.setAtFormatLength(8);
         tmess.setDataType("character");
         tmess.setFormat("-zzzzzz9");
         tmess.setLabel("tmess");
```

In 4207a:

```
         frame.setDown(1);
         tmess.setAt(true);
         tmess.setAtFormatLength(8);
         tmess.setDataType("character");
         tmess.setLabel("tmess");
```

#33 - 01/20/2020 07:11 AM - Greg Shah

I suspect that the format in trunk was wrong. I think the conversion of the frame def looks OK here in 4207a.

What about the business logic? Does it have the "-zzzzzz9" passed to the display statement inside the related frame element?

#34 - 01/20/2020 08:56 AM - Roger Borrello

Greg Shah wrote:

> I suspect that the format in trunk was wrong. I think the conversion of the frame def looks OK here in 4207a.
>
> What about the business logic? Does it have the "-zzzzzz9" passed to the display statement inside the related frame element?

The business logic converts the same way in both trunk and 4207a:

```
public class AtBaseTypeMismatch
{
   AtBaseTypeMismatchF1 f1Frame = GenericFrame.createFrame(AtBaseTypeMismatchF1.class, "f1");

   /**
    * External procedure (converted to Java from the 4GL source code
    * in at/at_base_type_mismatch.p).
    */
   public void execute()
   {
      decimal mymess = TypeFactory.decimal(new decimal(1.00));
      character tmess = UndoableFactory.character();

      externalProcedure(AtBaseTypeMismatch.this, new Block((Body) () ->
      {
         f1Frame.openScope();

         FrameElement[] elementList0 = new FrameElement[]
         {
            new Element(new integer(mymess), "-zzzzzz9", f1Frame.widgetTmess())
         };
          /* Dummy var used in @ */
         f1Frame.display(elementList0);

         message("Done.");
      }));
   }
}
```

#35 - 01/20/2020 09:45 AM - Greg Shah

This is OK.

#36 - 01/20/2020 11:49 AM - Roger Borrello

Greg Shah wrote:

> This is OK.

Good... moving onto adding the annotations in frame_scoping.rules to allow setAtFormatLength() to be created, without utilizing the widget list in convert/frame_generator.xml...

I plan on updating the add_widget function to place an annotation when we decide to not place the @ in the widget list. The determination that we are already an @ base-field with a sibling format phrase has been determined. I was going to use formattedResultSize() from com.goldencode.p2j.convert.ExpressionConversionWorker to get the length to add as an annotation.

My tree looks like (this=12884901925):

```
     [java] display [KW_DISP]:12884901923 @3:1
     [java]    expression [EXPRESSION]:12884901925 @0:0
     [java]       int [FUNC_INT]:12884901926 @3:9
     [java]          mymess [VAR_DEC]:12884901928 @3:13
     [java]    format phrase [FORMAT_PHRASE]:12884901931 @0:0
     [java]       format [KW_FORMAT]:12884901932 @3:21
     [java]          expression [EXPRESSION]:12884901934 @0:0
     [java]             "-zzzzzz9" [STRING]:12884901935 @3:28
     [java]       @ [AT]:12884901937 @3:40
     [java]          tmess [VAR_CHAR]:12884901939 @3:42
     [java]    with [FRAME_PHRASE]:12884901941 @3:48
     [java]       frame [KW_FRAME]:12884901943 @3:53
     [java]          f1 [WID_FRAME]:12884901945 @3:59
```

at the point I retrieve ref_fmtp = #(com.goldencode.ast.Aast) execLib("get_format_phrase", this).

ref_at = ref_fmtp.getImmediateChild(prog.at, null) not returning null has been my indicator to **not** add to the widget list, so I wanted to augment the rule to find the format expression. However, my ref_fmt = ref_fmtp.getImmediateChild(prog.kw_format,null) creates an expression execution error:

```
     [java] EXPRESSION EXECUTION ERROR:
     [java] ---------------------------
     [java] evalLib("add_widget")
     [java] ^  { Expression error [ref_fmt = ref_fmtp.getImmediateChild(prog.kw_format, null] }
     [java] ---------------------------
     [java] ERROR:
     [java] com.goldencode.p2j.pattern.TreeWalkException: ERROR!  Active Rule:
     [java] -----------------------
     [java]       RULE REPORT
     [java] -----------------------
     [java] Rule Type :   WALK
     [java] Source AST:  [ expression ] BLOCK/STATEMENT/KW_DISP/EXPRESSION/ @0:0 {12884901925}
     [java] Copy AST  :  [ expression ] BLOCK/STATEMENT/KW_DISP/EXPRESSION/ @0:0 {12884901925}
     [java] Condition :  ref_fmt = ref_fmtp.getImmediateChild(prog.kw_format, null
     [java] Loop      :  false
     [java] --- END RULE REPORT ---
```

I do see 2 for getNumImmediateChildren() but I'm still handing from the tree, instead of climbing it.

Does this seem like I'm on the right track?

#37 - 01/20/2020 01:42 PM - Roger Borrello

Roger Borrello wrote:

> Greg Shah wrote:
>
>> This is OK.
>
> Good... moving onto adding the annotations in frame_scoping.rules to allow setAtFormatLength() to be created, without utilizing the widget list in convert/frame_generator.xml...
>
> I plan on updating the add_widget function to place an annotation when we decide to not place the @ in the widget list. The determination that we are already an @ base-field with a sibling format phrase has been determined. I was going to use formattedResultSize() from com.goldencode.p2j.convert.ExpressionConversionWorker to get the length to add as an annotation.
>
> My tree looks like (this=12884901925):
> [...]
>
> at the point I retrieve ref_fmtp = #(com.goldencode.ast.Aast) execLib("get_format_phrase", this).
>
> ref_at = ref_fmtp.getImmediateChild(prog.at, null) not returning null has been my indicator to **not** add to the widget list, so I wanted to augment the rule to find the format expression. However, my ref_fmt = ref_fmtp.getImmediateChild(prog.kw_format,null) creates an expression execution error:
> [...]
>
> I do see 2 for getNumImmediateChildren() but I'm still handing from the tree, instead of climbing it.
>
> Does this seem like I'm on the right track?

Typo in the variable declaration <blush/>

#38 - 01/20/2020 01:43 PM - Greg Shah

> [java] Condition : ref_fmt = ref_fmtp.getImmediateChild(prog.kw_format, null

Do you have a missing closing parenthesis?

> Does this seem like I'm on the right track?

Yes, if you are doing this in the add_widget in frame_scoping.rules. Doing it in frame_generator.xml is too late.

The other thing I wonder about is how we calculate the longest format length. It isn't necessarily at this given location. We have to check all the @ base field cases. I think we already do that somewhere. Where is it and how do we access that value later?

#39 - 01/20/2020 02:15 PM - Roger Borrello

Greg Shah wrote:

> > [java] Condition : ref_fmt = ref_fmtp.getImmediateChild(prog.kw_format, null
>
> Do you have a missing closing parenthesis?

Post hoc ergo propter hoc. It was a misspelled variable name.

> > Does this seem like I'm on the right track?
>
> Yes, if you are doing this in the add_widget in frame_scoping.rules. Doing it in frame_generator.xml is too late.

I'm planning on retrieving the annotation in frame_generator.xml (at this time).

> The other thing I wonder about is how we calculate the longest format length. It isn't necessarily at this given location. We have to check all the @ base field cases. I think we already do that somewhere. Where is it and how do we access that value later?

This is where frame_generator.xml goes *Wizard of Oz* with the widget and options lists. It's very hard to follow. The main **walk-rule** performs *format string "preprocessing" which occurs for every instance of each form of a node that would by itself create a new frame field)*

```
            ((type == prog.expression and
              parent.type != prog.frame_element)   or
             evalLib("literals", type)             or
             (evalLib("fieldtype", type) and
              parent.type != prog.expression)      or
             (evalLib("vartype", type) and
              parent.type != prog.expression)      or
             type == prog.symbol)                      and
            isNote("frame-id")                         and
            parent.type != prog.frame_element
```

2 functions come into play at this point, get_explicit_format_string and get_override_format_string, both from common-progress.rules. Those functions, especially the override, seem to determine precedence:

```
      <!-- Find or build the format string for a given expression that will
           be displayed in a base field (the node passed MUST have a
           following format phrase with an AT node).  The following
           precedence order will be enforced:

           1. an explicit format string in a format phrase (but only in the
              case that such a format string occurs BEFORE an @ base field
              clause, otherwise the format string applies to that base field
              rather than to the given node)
           2. if the expression is a string literal, a manufactured format
              large enough to display the entire string will be returned
           3. if the data type of the expression is the same as that of the
              base field, then a null format string will be returned
           4. if the widget is a variable, an explicit format string in the
              var def
           5. if the widget is a field, an explicit format string in the
              field def (in the schema)
           6. the default format string for that data type (this includes
              literals which generally have the same default format string
              as the associated data type), even if the widget is a
              "complex expression" (not a var, field or literal)

           Please note that in the case that a non-override field ref is
           passed to this function, the result will NOT follow the rules
           above, but will instead be equivalent to the "get_format_string"
           function.
           -->
```

#40 - 01/20/2020 06:14 PM - Roger Borrello

My annotations are being put into the tree:

```
          <ast col="40" id="12884901937" line="3" text="@" type="AT">
            <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
            <annotation datatype="java.lang.Long" key="at-format-length" value="8"/>
            <ast col="42" id="12884901939" line="3" text="tmess" type="VAR_CHAR">
              <annotation datatype="java.lang.Long" key="oldtype" value="2782"/>
              <annotation datatype="java.lang.Long" key="refid" value="12884901909"/>
              <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
              <annotation datatype="java.lang.String" key="getter" value="getTmess"/>
              <annotation datatype="java.lang.String" key="setter" value="setTmess"/>
              <annotation datatype="java.lang.String" key="widgettype" value="FillInWidget"/>
              <annotation datatype="java.lang.String" key="javaname" value="tmess"/>
              <annotation datatype="java.lang.String" key="accessor" value="widgetTmess"/>
              <annotation datatype="java.lang.Boolean" key="firstref" value="true"/>
              <annotation datatype="java.lang.Boolean" key="lastref" value="true"/>
              <annotation datatype="java.lang.Long" key="frame-id" value="12884901956"/>
            </ast>
          </ast>
```

My logic added to add_widget is below. I need to think about the usefulness of the check for an existing annotation. I don't think they are re-used.

```
            <!-- Look for an @ base-field which should suppress the generation of a widget -->
            <action>fmtph = execLib("get_format_phrase", copy)</action>
            <rule>fmtph != null
               <action>ref_at = fmtph.getImmediateChild(prog.at, null)</action>
               <rule>ref_at != null
                  <action>do_add = false</action>

                  <action>fmtst = execLib("get_override_format_string", copy)</action>
                  <rule>fmtst != null
                     <action>fmtlen = ecw.formattedResultSize(this, fmtst, true)</action>
                     <action>curlen = ref_at.getAnnotation("at-format-length")</action>
                     <rule>curlen != null
                        <action on="false">ref_at.putAnnotation("at-format-length", #(java.lang.Long) fmtlen)<
/action>
                        <rule>fmtlen &gt; curlen
                           <action>ref_at.putAnnotation("at-format-length", #(java.lang.Long) fmtlen)</action>
                        </rule>
                     </rule>
                  </rule>
               </rule>
            </rule>
```

The retrieval end of the annotation needs some deep thought.

#41 - 01/23/2020 07:43 PM - Roger Borrello

- % Done changed from 100 to 90

- Status changed from Test to WIP

- Assignee set to Roger Borrello

#42 - 01/23/2020 07:58 PM - Roger Borrello

I have been working with frame_generator.xml and common-progress.rules to correct some regressions. I have a testcase which combines many of the scenarios causing issues (uast/at/at_frame_broken_display6-standalone.p).

The majority of differences between trunk and 4207a frame definitions are due to the fact that we had been creating widgets for expressions on the left side of the @ including annotations that were relied upon by some common-progress.rules functions, especially those looking for format phrases.

I have a specific question about get_explicit_format_string.

```
         <!-- obscure case where the frame field being processed is an
              @ base field reference (so it is itself inside a format
              phrase) -->
         <rule>
            fmtph == null                and
            ref.parent.type == prog.at   and
            ref.parent.parent.type == prog.format_phrase

            <action>fmtph = ref.parent.parent</action>
            <!-- check for a FOLLOWING format string but ignore any that
                 appear before our @ node parent because the 4GL assigns
                 such format strings to the overriding field not the base
                 field -->
            <action>
<!--              fmtst = fmtph.getImmediateChild(prog.kw_format, ref.parent)-->
               fmtst = fmtph.getImmediateChild(prog.kw_format, null)
            </action>
         </rule>
```

By utilizing the new line below the commented line, the format string is found correctly in our new tree. What does the ref.parent changed to null to the getImmediateChild?

#43 - 01/24/2020 08:19 AM - Greg Shah

Did you read the javadoc?

#44 - 01/24/2020 09:34 AM - Roger Borrello

- % Done changed from 90 to 100

- Assignee deleted (Roger Borrello)

Greg Shah wrote:

> Did you read the javadoc?

Got it. Thanks for the discussion.

Regarding "junk" before the @ base-field, 4GL is pretty explicit when you try to put stuff in:
** Only WHEN and FORMAT allowed before @ phrase. (579)

The WHEN and FORMAT can be in any order:
display myint when myint = 9 format "-zzzzzzz9" @ t-int with frame f1. or display myint format "-zzzzzzz9" when myint = 9 @ t-int with frame f1. both are OK.

#45 - 01/24/2020 12:13 PM - Roger Borrello

I wanted to check that this is a valid addition to the frame definition for this testcase:

```
/* ADD FIELD "addr-id" OF "Pers-Addr" AS integer FORMAT "999999" LABEL "Addr ID" */
display " " @ Address.addr-id with frame f1.
```

These are added to the frame definition when compared to trunk:

```
         addrId.setDbname("p2j_test");
         addrId.setTable("address");
         addrId.setDataType("integer");
```

The other fields options were already in the definition:

```
         addrId.setAt(true);
         addrId.setAtFormatLength(6);
         addrId.setFormat("999999");
```

It seems like having those would be pretty important, but they aren't in the trunk version.

#46 - 01/24/2020 12:44 PM - Greg Shah

> > addrId.setAt(true);
> > addrId.setAtFormatLength(6);

I'm not sure about these. You'll have to look at how we use them in the runtime code. However, I suspect these are only needed if the override needs to be larger than the widget format.

> addrId.setFormat("999999");

This one needs to be there. I'm very surprised it isn't there in trunk.

#47 - 01/24/2020 01:35 PM - Roger Borrello

Greg Shah wrote:

> > addrId.setFormat("999999");
>
> This one needs to be there. I'm very surprised it isn't there in trunk.

It was... those last 3 I mentioned were in trunk **and** 4207a.

#48 - 01/24/2020 01:45 PM - Roger Borrello

Roger Borrello wrote:

> I wanted to check that this is a valid addition to the frame definition for this testcase:
>
> [...]
>
> These are added to the frame definition when compared to trunk:
> [...]

To clarify the question, the below 3 lines were added now in **4207a** and not in **trunk**. I am not sure why at this point. Other than that, the frame defs were the same.

```
         addrId.setDbname("p2j_test");
         addrId.setTable("address");
         addrId.setDataType("integer");
```

#49 - 01/24/2020 04:47 PM - Greg Shah

It is more correct now. The reason is probably simple: the widget before was being created from a string literal and now the widget definition is a FIELD_INT. Widgets defined from fields need some database info so that some attributes can be mapped properly.

#50 - 01/29/2020 03:51 PM - Roger Borrello

Found that since we have reverted code, and the widget list for frame f2 now contains the variable that was before the @ base-field, the search mechanism looking for the f1 widget finds it on f2, resulting in the wrong frame-id being annotated under the INPUT node.

You mentioned keeping a stack, and that I could find an example in control-flow.rules, but I don't see any Dictionary usage in there. Is there another implementation I should use, instead of following the frame_scoping.rules model?

#51 - 01/29/2020 03:53 PM - Roger Borrello

- % Done changed from 100 to 90

#52 - 01/29/2020 04:05 PM - Greg Shah

convert/control_flow.rules, search for Dictionary.

#53 - 01/30/2020 04:26 PM - Roger Borrello

Greg Shah wrote:

> convert/control_flow.rules, search for Dictionary.

Thanks. That shows adding to the "switch" Dictionary in pairs (useSwitch, caseName) via consecutive calls to addDictionary___. I could do similar using:

```
      <!-- Upon entering an override frame block, to save state
           and update override variables -->
      <function name="enter_override_frame_block">
         <parameter name="frame_name" type="java.lang.String" />
         <parameter name="blockid"   type="java.lang.Long" />
         <rule>block-id &gt; 0
            <action>addDictionaryString("overrideFrame", override_frame_name)</action>
            <action>addDictionaryLong("overrideFrame", override_frame_blockid)</action>
            <action>override_frame_name    = frame_name</action>
            <action>override_frame_blockid = blockid</action>
         </rule>
      </function>
```

But I'm not sure that's the best way. What method is used to remove an item? I don't see any. What about a ScopedSymbolDictionary, where I add a scope then add symbols in the walk-rules, then delete in the ascent-rules?

Speaking of that, we discussed INNER_BLOCK/FRAME_PHRASE/KW_FRAME/WID_FRAME to indicate a new override frame. However, there are a few nodes between INNER_BLOCK and FRAME_PHRASE/KW_FRAME/WID_FRAME, such as REPEAT in our case. I could easily match that with INNER_BLOCK/REPEAT/FRAME_PHRASE/KW_FRAME when I get to type==prog.wid_frame, but what about other language statements such as DO and FOR? Is this a good way to cover that in the walk-rule?

```
      <rule>upPath("FRAME_PHRASE/KW_FRAME") and type == prog.wid_frame and
            (parent.parent.parent.type == prog.kw_repeat or
             parent.parent.parent.type == prog.kw_do or
             parent.parent.parent.type == prog.kw_for) and
            parent.parent.parent.parent.type == prog.inner_block
         <action>printfln("frame_scoping: potential new frame\n%s", this.parent.parent.parent.parent.dumpTree(
))</action>
      </rule>
```

I assume the ascent-rule would be similar, but upon leaving the INNER_BLOCK node, I'd have to look down for similar relationships. But from what I read, downPath is no efficient... should I leave the prog.wid_frame node, and look up?

#54 - 01/30/2020 04:48 PM - Greg Shah

> What about a ScopedSymbolDictionary, where I add a scope then add symbols in the walk-rules, then delete in the ascent-rules?

That same rule set has the push of a new scope (<action>addScope("switch")</action>) and the pop of a scope (<action>deleteScope("switch")</action>). The dictionaries are scoped naturally (they are a stack of maps where you decide when to push/pop the next scope).

> Is this a good way to cover that in the walk-rule?

Yes, it is OK. I probably would use relativePath("FRAME_PHRASE/KW_FRAME/WID_FRAME") and then from the type == prog.wid_frame and part, but it is 6 of one and a half-dozen of another.

> I assume the ascent-rule would be similar, but upon leaving the INNER_BLOCK node, I'd have to look down for similar relationships. But from what I read, downPath is no efficient... should I leave the prog.wid_frame node, and look up?

No. You are going to set override_frame_blockid to be the parent.parent.parent.parent.id. Then on ascent, you would have something like this:

```
<rule>type == prog.inner_block and id = override_frame_blockid
   RESTORE STATE AND THEN POP SCOPE
</rule>
```

#55 - 01/30/2020 06:00 PM - Roger Borrello

I have the following, but am trying to determine what type to make the ID, since we wanted to initialize it to -1. Right now, I have:

```
   <!-- Frame name and block ID that can override unspecified frame statements -->
   <variable name="override_frame_blockid"   type="java.lang.Integer" />
   <variable name="override_frame_name"      type="java.lang.String" />
   <variable name="override_frame_widnode"   type="com.goldencode.ast.Aast" />
   <variable name="override_frame_blocknode" type="com.goldencode.ast.Aast" />
```

For the init

```
      <!-- Information to use when no frame is explicitly defined -->
      <rule>deleteDictionary("overrideFrame")</rule>
      <rule>override_frame_blockid = -1</rule>
      <rule>override_frame_name = "" </rule>
```

For the pusher/popper:

```
      <!-- Upon entering an override frame block, to save state
           and update override variables -->
      <function name="enter_override_frame_block">
         <parameter name="frame_name" type="java.lang.String" />
         <parameter name="blockid"   type="java.lang.Long" />
         <rule>true
            <action>addScope("overrideFrame")</action>
            <action>addDictionaryString("overrideFrame", "frame", override_frame_name)</action>
            <action>addDictionaryLong("overrideFrame", "blockid", override_frame_blockid)</action>
            <action>override_frame_name    = frame_name</action>
            <action>override_frame_blockid = blockid</action>
         </rule>
      </function>

      <!-- Upon leaving an override frame block, to save state
           and update override variables -->
      <function name="leave_override_frame_block">
         <parameter name="frame_name" type="java.lang.String" />
         <parameter name="blockid"   type="java.lang.Long" />
         <rule>true
            <action>deleteScope("overrideFrame")</action>
            <action>override_frame_name    = lookupDictionaryBoolean("overrideFrame", "frame")</action>
            <action>override_frame_blockid = lookupDictionaryLong("overrideFrame", "blockid")</action>
         </rule>
      </function>
```

For the walk-rule

```
      <rule>relativePath("FRAME_PHRASE/KW_FRAME/WID_FRAME") and type == prog.wid_frame and 
            (parent.parent.parent.type == prog.kw_repeat or
             parent.parent.parent.type == prog.kw_do or
             parent.parent.parent.type == prog.kw_for) and
            parent.parent.parent.parent.type == prog.inner_block
         <action>override_frame_blocknode = copy.parent.parent.parent.parent</action>
         <action>override_frame_widnode = copy</action>
         <action>evalLib("enter_override_frame_block", override_frame_widnode.text, override_frame_blocknode.i
d)</action>
         <action>printfln("frame_scoping: potential new frame=%d,%s", 
                          override_frame_blocknode.id, override_frame_widnode.text)</action>
      </rule>
```

And for the ascent-rule

```
      <rule>type == prog.inner_block and id == override_frame_blockid
         <action>evalLib("leave_override_frame_block", override_frame_name, override_frame_blockid)</action>
         <action>printfln("frame_scoping: potential old frame=%d,%s", 
                          override_frame_blocknode.id, override_frame_widnode.text)</action>
      </rule>
```

#56 - 01/30/2020 06:20 PM - Greg Shah

> I have the following, but am trying to determine what type to make the ID, since we wanted to initialize it to -1.

It must be compatible with the type of id which is returned by AnnotatedAst.getId(). If you have a problem assigning -1 to that type, try casting it like #(the_type_of_id) (-1).

In leave_override_frame_block, why do you have parameters that are not used?

In leave_override_frame_block, the delete is out of order.

In the walk rule, you don't need type == prog.wid_frame and it is redundant.

Otherwise it is the right idea.

#57 - 01/31/2020 09:13 AM - Roger Borrello

Greg Shah wrote:

> > I have the following, but am trying to determine what type to make the ID, since we wanted to initialize it to -1.
>
> It must be compatible with the type of id which is returned by AnnotatedAst.getId(). If you have a problem assigning -1 to that type, try casting it like #(the_type_of_id) (-1).

#(java.lang.Long) (-1) did the trick. :-)

> In leave_override_frame_block, why do you have parameters that are not used?

Copy/past of the enter then brain fart.

> In leave_override_frame_block, the delete is out of order.

Thanks.

> In the walk rule, you don't need type == prog.wid_frame and it is redundant.
>
> Otherwise it is the right idea.

Yes, it works. The widgets are on the right frames.

Time for some regression testing.

#58 - 01/31/2020 09:54 AM - Roger Borrello

I did see a lot of "DANGER WILL ROBINSON!" messages on some regression tests yesterday from the search_frame_phrase function. I have changed the message to printfln("## WARNING: %s: Usingframe='%s' from Dictionary for ref='%s', id=%d", file, frame.get(frameName), ref.text, ref.id) and I will report on them.

#59 - 01/31/2020 09:56 AM - Greg Shah

> I have changed the message to printfln("## WARNING: %s: Usingframe='%s' from Dictionary for ref='%s', id=%d", file, frame.get(frameName), ref.text, ref.id) and I will report on them.

We need to know the up path for each of these scenarios to understand why/how it is being used.

#60 - 01/31/2020 10:00 AM - Roger Borrello

Greg Shah wrote:

> > I have changed the message to printfln("## WARNING: %s: Usingframe='%s' from Dictionary for ref='%s', id=%d", file, frame.get(frameName), ref.text, ref.id) and I will report on them.
>
> We need to know the up path for each of these scenarios to understand why/how it is being used.

I saw another that was printfln("## WARN: %s: no frame found for %s/%s/%s (%s)", file, parent.parent, parent, this, text). I can include ref.parent.parent, or whatever you suggest.

#61 - 01/31/2020 10:45 AM - Greg Shah

Optimally, we would see the dumpTree() from the ancestor STATEMENT node.

#62 - 01/31/2020 11:31 AM - Roger Borrello

Check my placement of the check:

Currently we call "search_widget":

```
                        <!-- look for widget in one of frames -->
                        <rule>tmp == null
                        <action on="false">ref3.putAnnotation("check_accessors", "")</action>
                        <action>tmp = execLib("search_widget", ref3)</action>
                     </rule>
```

In "search_widget" I'm updating the <rule>true to:

```
         <rule>override_frame_blockid != -1
            <action>res = execLib("get_named_frame", override_frame_name)</action>
         </rule>

         <rule>res == null
```

So if there is an override_frame_blockid, we'll use it.

#63 - 01/31/2020 12:06 PM - Greg Shah

search_widget is used in many places so we must be especially sure that your change is safe. I can't tell without seeing the full set of changes.

#64 - 01/31/2020 12:49 PM - Roger Borrello

Greg Shah wrote:

> search_widget is used in many places so we must be especially sure that your change is safe. I can't tell without seeing the full set of changes.

Committed revision 11386.

Review would be appreciated.

```
modified rules/annotations/frame_scoping.rules
modified rules/convert/frame_generator.xml
```

#65 - 01/31/2020 01:01 PM - Greg Shah

Code Review Task Branch 4207a Revision 11386

I'm OK with the changes. If it passes regression testing, while still resolving everything found in the testcases and the customer app, then we are good to go.

#66 - 01/31/2020 01:03 PM - Roger Borrello

Greg Shah wrote:

> Code Review Task Branch 4207a Revision 11386
>
> I'm OK with the changes. If it passes regression testing, while still resolving everything found in the testcases and the customer app, then we are good to go.

conv-regression running, and about to kick off some other customer conversion.

#67 - 01/31/2020 03:14 PM - Roger Borrello

Bug found when converting customer code.

Testcase:

```
def temp-table tt field cost as decimal format "->>,>>9.99" label "Cost".
def buffer ttbuff for tt.
create tt. tt.cost = 9.99.
def var mywid as character no-undo format "x(6)".
form mywid with frame f1.

do with frame f1:
  display buffer ttbuff:buffer-field("cost"):buffer-value @ mywid.
end.
message "Done." 
```

Checked in as: uast/at/at_frame_buffer-field.p

It leads to a conversion error:

```
     [java] ------------------------------------------------------------------------------
     [java] Code Conversion Annotations
     [java] ------------------------------------------------------------------------------
     [java] 
     [java] Optional rule set [customer_specific_annotations_prep] not found.
     [java] ./uast/at/at_frame_buffer-field.p
     [java] ## INFO: ./uast/at/at_frame_buffer-field.p: assuming DOWN set to 1 for f1
     [java] Elapsed job time:  00:00:01.291
     [java] EXPRESSION EXECUTION ERROR:
     [java] ---------------------------
     [java] throwException( sprintf("## unknown variable type %s/%s/%s", ref.parent.parent, ref.parent, ref))
     [java] ^  { ## unknown variable type COLON/COLON/KW_BUFFER [METH_HANDLE id <12884901997> 8:25] }
     [java] ---------------------------
     [java] EXPRESSION EXECUTION ERROR:
```

```
     [java] ---------------------------
     [java] evalLib("gen_accessors", ref1, tmp)
     [java] ^  { Expression execution error @1:1 }
     [java] ---------------------------
     [java] ERROR:
     [java] com.goldencode.p2j.pattern.TreeWalkException: ERROR!  Active Rule:
     [java] -----------------------
     [java]       RULE REPORT
     [java] -----------------------
     [java] Rule Type :   WALK
     [java] Source AST:  [ buffer-field ] BLOCK/INNER_BLOCK/BLOCK/STATEMENT/KW_DISP/EXPRESSION/COLON/COLON/MET
H_HANDLE/ @8:25 {12884901997}
     [java] Copy AST  :  [ buffer-field ] BLOCK/INNER_BLOCK/BLOCK/STATEMENT/KW_DISP/EXPRESSION/COLON/COLON/MET
H_HANDLE/ @8:25 {12884901997}
     [java] Condition :  throwException( sprintf("## unknown variable type %s/%s/%s", ref.parent.parent, ref.p
arent, ref))
     [java] Loop      :  false
     [java] --- END RULE REPORT ---
     [java] 
     [java] 
     [java] 
     [java]     at com.goldencode.p2j.pattern.PatternEngine.run(PatternEngine.java:1070)
     [java]     at com.goldencode.p2j.convert.TransformDriver.processTrees(TransformDriver.java:542)
     [java]     at com.goldencode.p2j.convert.ConversionDriver.back(ConversionDriver.java:562)
     [java]     at com.goldencode.p2j.convert.TransformDriver.executeJob(TransformDriver.java:876)
     [java]     at com.goldencode.p2j.convert.ConversionDriver.main(ConversionDriver.java:983)
     [java] Caused by: com.goldencode.expr.ExpressionException: Expression execution error @1:1 [METH_HANDLE i
d=12884901997]
     [java]     at com.goldencode.p2j.pattern.AstWalker.walk(AstWalker.java:275)
     [java]     at com.goldencode.p2j.pattern.AstWalker.walk(AstWalker.java:210)
     [java]     at com.goldencode.p2j.pattern.PatternEngine.apply(PatternEngine.java:1633)
     [java]     at com.goldencode.p2j.pattern.PatternEngine.processAst(PatternEngine.java:1531)
     [java]     at com.goldencode.p2j.pattern.PatternEngine.processAst(PatternEngine.java:1479)
     [java]     at com.goldencode.p2j.pattern.PatternEngine.run(PatternEngine.java:1034)
     [java]     ... 4 more
     [java] Caused by: com.goldencode.expr.ExpressionException: Expression execution error @1:1
     [java]     at com.goldencode.expr.Expression.execute(Expression.java:484)
     [java]     at com.goldencode.p2j.pattern.Rule.apply(Rule.java:497)
     [java]     at com.goldencode.p2j.pattern.Rule.executeActions(Rule.java:745)
     [java]     at com.goldencode.p2j.pattern.Rule.coreProcessing(Rule.java:712)
     [java]     at com.goldencode.p2j.pattern.Rule.apply(Rule.java:534)
     [java]     at com.goldencode.p2j.pattern.Rule.executeActions(Rule.java:745)
     [java]     at com.goldencode.p2j.pattern.Rule.coreProcessing(Rule.java:712)
     [java]     at com.goldencode.p2j.pattern.Rule.apply(Rule.java:534)
     [java]     at com.goldencode.p2j.pattern.RuleContainer.apply(RuleContainer.java:585)
     [java]     at com.goldencode.p2j.pattern.RuleSet.apply(RuleSet.java:98)
     [java]     at com.goldencode.p2j.pattern.AstWalker.walk(AstWalker.java:262)
     [java]     ... 9 more
     [java] Caused by: com.goldencode.expr.ExpressionException: Expression execution error @1:1
     [java]     at com.goldencode.expr.Expression.execute(Expression.java:484)
     [java]     at com.goldencode.p2j.pattern.Rule.apply(Rule.java:497)
     [java]     at com.goldencode.p2j.pattern.Rule.executeActions(Rule.java:745)
     [java]     at com.goldencode.p2j.pattern.Rule.coreProcessing(Rule.java:712)
     [java]     at com.goldencode.p2j.pattern.Rule.apply(Rule.java:534)
     [java]     at com.goldencode.p2j.pattern.Rule.executeActions(Rule.java:745)
     [java]     at com.goldencode.p2j.pattern.Rule.coreProcessing(Rule.java:712)
     [java]     at com.goldencode.p2j.pattern.Rule.apply(Rule.java:534)
     [java]     at com.goldencode.p2j.pattern.NamedFunction.execute(NamedFunction.java:450)
     [java]     at com.goldencode.p2j.pattern.AstSymbolResolver.execute(AstSymbolResolver.java:712)
     [java]     at com.goldencode.p2j.pattern.CommonAstSupport$Library.execLib(CommonAstSupport.java:1534)
     [java]     at com.goldencode.p2j.pattern.CommonAstSupport$Library.evalLib(CommonAstSupport.java:1510)
     [java]     at com.goldencode.expr.CE8961.execute(Unknown Source)
     [java]     at com.goldencode.expr.Expression.execute(Expression.java:391)
     [java]     ... 19 more
     [java] Caused by: com.goldencode.p2j.pattern.CommonAstSupport$UserGeneratedException: ## unknown variable
 type COLON/COLON/KW_BUFFER [METH_HANDLE id <12884901997> 8:25]
     [java]     at com.goldencode.p2j.pattern.CommonAstSupport$Library.throwException(CommonAstSupport.java:28
17)
     [java]     at com.goldencode.p2j.pattern.CommonAstSupport$Library.throwException(CommonAstSupport.java:28
02)
     [java]     at com.goldencode.expr.CE8967.execute(Unknown Source)
     [java]     at com.goldencode.expr.Expression.execute(Expression.java:391)
     [java]     ... 32 more

BUILD FAILED
```

#68 - 01/31/2020 06:21 PM - Roger Borrello

While that testcase does not throw the exception in trunk, it still reports issues. I added some debug to my trunk copy:

```
     [java] ------------------------------------------------------------------------------
     [java] Frame Generator
     [java] ------------------------------------------------------------------------------
     [java]
     [java] ./uast/at/at_frame_buffer-field.p
     [java] common-progress.get_explicit_format_string (trunk):
     [java] expression [EXPRESSION]:12884901966 @0:0
     [java]    mywid [VAR_CHAR]:12884901967 @5:6
     [java]
     [java] common-progress.get_explicit_format_string (trunk): return fmt=null
     [java] common_progress.get_override_format_string (trunk): base=null
     [java] expression [EXPRESSION]:12884901966 @0:0
     [java]    mywid [VAR_CHAR]:12884901967 @5:6
     [java]
     [java] common-progress.get_explicit_format_string (trunk):
     [java] expression [EXPRESSION]:12884901966 @0:0
     [java]    mywid [VAR_CHAR]:12884901967 @5:6
     [java]
     [java] common-progress.get_explicit_format_string (trunk): return fmt=null
     [java] common_progress.get_override_format_string (trunk): returning fmt=x(6)
     [java] common-progress.get_explicit_format_string (trunk):
     [java] expression [EXPRESSION]:12884901990 @0:0 HIDDEN
     [java]    : [COLON]:12884901991 @8:45
     [java]       : [COLON]:12884901993 @8:24
     [java]          buffer [KW_BUFFER]:12884901994 @8:11
     [java]             ttbuff [BUFFER]:12884901996 @8:18
     [java]          buffer-field [METH_HANDLE]:12884901997 @8:25
     [java]             "cost" [STRING]:12884901999 @8:38
     [java]       buffer-value [ATTR_POLY]:12884902000 @8:46
     [java]
     [java] common-progress.get_explicit_format_string (trunk): return fmt=null
     [java] common_progress.get_override_format_string (trunk): base != null
     [java] @ [AT]:12884902003 @8:59
     [java]    mywid [VAR_CHAR]:12884902005 @8:61
     [java]
     [java] common-progress.get_explicit_format_string (trunk):
     [java] expression [EXPRESSION]:12884901990 @0:0 HIDDEN
     [java]    : [COLON]:12884901991 @8:45
     [java]       : [COLON]:12884901993 @8:24
     [java]          buffer [KW_BUFFER]:12884901994 @8:11
     [java]             ttbuff [BUFFER]:12884901996 @8:18
     [java]          buffer-field [METH_HANDLE]:12884901997 @8:25
     [java]             "cost" [STRING]:12884901999 @8:38
     [java]       buffer-value [ATTR_POLY]:12884902000 @8:46
     [java]
     [java] common-progress.get_explicit_format_string (trunk): return fmt=null
     [java] common-progress.get_override_format_string (trunk): 3
     [java] WARNING: no type calculated for BUFFER-VALUE in [12884902000] buffer-value [ATTR_POLY]:12884902000
 @8:46
     [java]
     [java] common_progress.get_override_format_string (trunk): base=null
     [java] expression [EXPRESSION]:12884901990 @0:0 HIDDEN
     [java]    : [COLON]:12884901991 @8:45
     [java]       : [COLON]:12884901993 @8:24
     [java]          buffer [KW_BUFFER]:12884901994 @8:11
     [java]             ttbuff [BUFFER]:12884901996 @8:18
     [java]          buffer-field [METH_HANDLE]:12884901997 @8:25
     [java]             "cost" [STRING]:12884901999 @8:38
     [java]       buffer-value [ATTR_POLY]:12884902000 @8:46
     [java]
     [java] common-progress.get_explicit_format_string (trunk):
     [java] expression [EXPRESSION]:12884901990 @0:0 HIDDEN
     [java]    : [COLON]:12884901991 @8:45
     [java]       : [COLON]:12884901993 @8:24
     [java]          buffer [KW_BUFFER]:12884901994 @8:11
     [java]             ttbuff [BUFFER]:12884901996 @8:18
     [java]          buffer-field [METH_HANDLE]:12884901997 @8:25
     [java]             "cost" [STRING]:12884901999 @8:38
```

```
    [java]       buffer-value [ATTR_POLY]:12884902000 @8:46
    [java] 
    [java] common-progress.get_explicit_format_string (trunk): return fmt=null
    [java] WARNING: no type calculated for BUFFER-VALUE in [12884902000] buffer-value [ATTR_POLY]:12884902000
@8:46
    [java] 
    [java] ISSUE #04: _POLY format at conversion time (cls is null)
    [java] common_progress.get_override_format_string (trunk): 4 base=null, fmt=x(8)
    [java] common_progress.get_override_format_string (trunk): returning fmt=x(8)
    [java] WARNING: no type calculated for BUFFER-VALUE in [12884902000] buffer-value [ATTR_POLY]:12884902000
@8:46
    [java] 
    [java] common-progress.get_explicit_format_string (trunk):
    [java] mywid [VAR_CHAR]:12884902005 @8:61
    [java] 
    [java] common-progress.get_explicit_format_string (trunk): obscure fmtph==null
    [java] common-progress.get_explicit_format_string (trunk): return fmt=null
    [java] common_progress.get_override_format_string (trunk): base=null
    [java] mywid [VAR_CHAR]:12884902005 @8:61
    [java] 
    [java] common-progress.get_explicit_format_string (trunk):
    [java] mywid [VAR_CHAR]:12884902005 @8:61
    [java] 
    [java] common-progress.get_explicit_format_string (trunk): obscure fmtph==null
    [java] common-progress.get_explicit_format_string (trunk): return fmt=null
    [java] common-progress.get_override_format_string (trunk): returning fmt=x(6)
    [java] common-progress.get_explicit_format_string (trunk):
    [java] mywid [VAR_CHAR]:12884902005 @8:61
    [java] 
    [java] common-progress.get_explicit_format_string (trunk): obscure fmtph==null
    [java] common-progress.get_explicit_format_string (trunk): return fmt=null
    [java] WARNING: no type calculated for BUFFER-VALUE in [12884902000] buffer-value [ATTR_POLY]:12884902000
@8:46
```

#69 - 01/31/2020 07:24 PM - Roger Borrello

Made an update to the testcase.

```
def temp-table tt field cost as decimal format "->>,>>9.99" label "Cost".
def buffer ttbuff for tt.
create tt. tt.cost = 9.99.
```

```
def var mywid as character no-undo format "x(6)".
def var mymess as char.
form mywid with frame f1.
form mymess with frame f2.

for each ttbuff:
   do with frame f1:
      /* The below line throws an exception in trunk and 4207a */
      //display buffer ttbuff:buffer-field("cost"):buffer-value @ mymess with frame f2.
      /* The below line throws an exception in 4207a */
      display buffer ttbuff:buffer-field("cost"):buffer-value @ mywid.
   end.
end.
message "Done."
```

#70 - 02/01/2020 10:53 AM - Roger Borrello

This issue was with where I had located the check for the override frame in search_widget. It needed to be used for the wlist search.

```
Committing to: /home/rfb/secure/code/p2j_repo/p2j/active/4207a/

modified rules/annotations/frame_scoping.rules
Committed revision 11387.
```

#71 - 02/01/2020 09:55 PM - Roger Borrello

Found another regression with this testcase:

```
def var cList as char view-as selection-list sort multiple inner-chars 26 inner-lines 8.
def var cDevices as char no-undo init "device1,device2".
form cList with frame frDevices.
update cList with frame frDevices.
def var i as int no-undo.
cList:list-items = cDevices.
do i = 1 to num-entries(cList:screen-value) with frame frUnlock row 5 overlay:
  display 'Unlocking: ' entry(i,cList:screen-value) format 'x(20)'.
end.
message "Done.".
```

The frUnlock is found to be an override frame, but there are no widgets associated with it. Checking how trunk makes the determination that cList is a widget on @frUnlock.

#72 - 02/03/2020 11:32 AM - Roger Borrello

Roger Borrello wrote:

> Found another regression with this testcase:
> [...]
>
> The frUnlock is found to be an override frame, but there are no widgets associated with it. Checking how trunk makes the determination that cList is a widget on @frUnlock.

The new override frame was being used exclusively, and not allowing other frames to be checked. Updated code, but currently looking over an issue using a browse widget.

#73 - 02/03/2020 12:55 PM - Roger Borrello

In looking at the new issue related to 4207a, it is not due to these changes related to override_frame, since I have modified frame_scoping.rules to the point where it is essentially behaving the same as trunk, yet we are left with several nodes missing annotations:

```
     [java] WARNING: Null annotation (frame-id) for <table>.<field> [FIELD_DEC] @6258:26 (17179902573)
     [java] WARNING: Null annotation (getter) for <table>.<field> [FIELD_DEC] @6258:26 (17179902573)
     [java] WARNING: Null annotation (setter) for <table>.<field> [FIELD_DEC] @6258:26 (17179902573)
     [java] WARNING: Null annotation (accessor) for <table>.<field> [FIELD_DEC] @6258:26 (17179902573)
     [java] WARNING: Null annotation (widgettype) for <table>.<field> [FIELD_DEC] @6258:26 (17179902573)
```

<table>.<field> redacted.

There may be a regression in another update to 4207a, but it's not in code related to 4476.

#74 - 02/03/2020 01:02 PM - Greg Shah

Is this related to embedded assignment?

#75 - 03/03/2020 02:24 PM - Greg Shah

Task branch 4207a was merged to trunk as revision 11344.

Is there anything left open in this task?

#76 - 03/03/2020 02:26 PM - Roger Borrello

Greg Shah wrote:

> Task branch 4207a was merged to trunk as revision 11344.
>
> Is there anything left open in this task?

Nothing that I know of.

#77 - 03/03/2020 02:36 PM - Greg Shah

- Status changed from WIP to Closed

- % Done changed from 90 to 100