User Interface - Feature #3856

implement a widget that replaces the Microsoft ImageList OCX (VB 6.0 common control)

01/10/2019 12:19 PM - Greg Shah

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

Description

Related issues:

Related to User Interface - Feature #1820: implement a tree widget to replace...	**New**
Related to User Interface - Feature #4174: implement calendar control/dtpicke...	**Closed**

History

#1 - 01/10/2019 12:28 PM - Greg Shah

See https://docs.microsoft.com/en-us/windows/desktop/controls/image-lists for documentation about the control.

#2 - 06/12/2019 04:29 AM - Hynek Cihlar

- Precedes Feature #3857: implement a widget that replaces the Microsoft ListView OCX (VB 6.0 common control) added

#3 - 06/12/2019 04:42 AM - Hynek Cihlar

I added a "Precedes" link to #3857, since ListView control has the ability to consume ImageList instances and use it to display images. However this feature doesn't seem to be used by any of the current large apps. So I'm removing the dependency. Btw. ImageList is used on several places with TreeView control.

#4 - 06/12/2019 04:42 AM - Hynek Cihlar

- Precedes deleted (Feature #3857: implement a widget that replaces the Microsoft ListView OCX (VB 6.0 common control))

#5 - 06/12/2019 07:12 AM - Greg Shah

> However this feature doesn't seem to be used by any of the current large apps. So I'm removing the dependency.

Yes, this makes sense. We can add that feature later.

#6 - 06/12/2019 08:08 AM - Hynek Cihlar

- Status changed from New to WIP

#8 - 06/12/2019 02:45 PM - Hynek Cihlar

Created task branch 3856a.

#9 - 06/14/2019 11:38 AM - Hynek Cihlar

3856a rebased against latest trunk.

#10 - 06/14/2019 05:26 PM - Hynek Cihlar

Task branch 3856a contains the initial implementation of IMAGELIST widget with the rules to convert the ImageList OCX.

Please review.

#11 - 06/14/2019 08:59 PM - Greg Shah

Code Review Task Branch 3856a Revision 11323

I'm good with the changes.

#12 - 08/22/2019 10:29 AM - Greg Shah

- Assignee set to Hynek Cihlar

Did these changes ever get merged into another branch or the trunk?

#13 - 08/22/2019 10:30 AM - Greg Shah

- Related to Feature #1820: implement a tree widget to replace the TreeView and ImageList OCXs added

#14 - 08/22/2019 12:04 PM - Hynek Cihlar

Greg Shah wrote:

> Did these changes ever get merged into another branch or the trunk?

No, these changes have not been merged. This issue precedes #4112 and I wanted to deliver this with that issue.

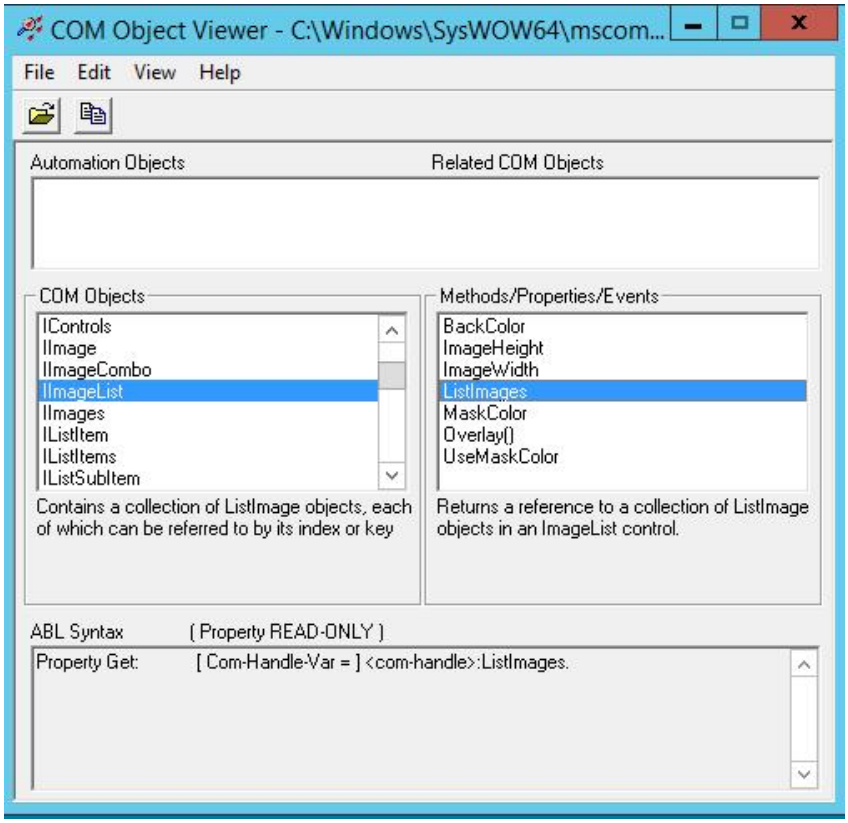
#15 - 11/01/2019 12:09 PM - Marius Gligor

- Assignee changed from Hynek Cihlar to Marius Gligor

- File abl-com-object-viewer.png added

ImageList OCX details

Unfortunately we cannot use the commons controls from mscomctl.ocx to create OE uast tests, due to missing the require license.
However, using OpenEdge Tools/PRO*Tools/COM Object Viewer by opening the mscomctl.ocx file, I extracted the following information regarding ImageList OCX.



This tool is useful because it describe the properties and methods of a COM interface using ABL syntax.

```
IImageList: (ImageList)
Contains a collection of ListImage objects each of which can be referred by its index or key.

Returns/sets the background color used to display text and graphics in an object.
Property Get:    [ Integer-Var = ] <com-handle>:BackColor.
Property Set:    <com-handle>:BackColor [ = Integer-Var ].

Returns/sets the height of a ListImage (IImage) object
Property Get:    [ Integer-Var = ] <com-handle>:ImageHeight.
Property Set:    <com-handle>:ImageHeight [ = Integer-Var ].

Returns/sets the width of a ListImage (IImage) object
Property Get:    [ Integer-Var = ] <com-handle>:ImageWidth.
Property Set:    <com-handle>:ImageWidth [ = Integer-Var ].

Returns a reference to a collection of ListImage object in an ImageList control (IImages).
Property Get:    [ Com-Handle-Var = ] <com-handle>:ListImages.

Returns/sets a value which determine the color to be transparent in ImageList graphical operation.
Property Get:    [ Integer-Var = ] <com-handle>:MaskColor.
Property Set:    <com-handle>:MaskColor [ = Integer-Var ].

Create a composite third image out of two ListImage and returns a reference to he new object.
[ <user-defined>-Var = ] <com-handle>: Overlay (
      <anytype>-Key1 BY-VARIANT-POINTER,
      <anytype>-Key2 BY-VARIANT-POINTER ).

Retuns/sets a value which determines if the ImageList control will use MaskColor property.
Property Get:    [ Logical-Var = ] <com-handle>:UseMaskColor.
Property Set:    <com-handle>:UseMaskColor [ = Logical-Var ].



IImages (ListImages)
A bitmap or icon of any size that can be used in others controls.

Adds a ListImage object to a ListImages and returns a reference to the created object (IImage).
[ Com-Handle-Var = ] <com-handle>: Add (
      <anytype>-Index BY-VARIANT-POINTER,
      <anytype>-Key BY-VARIANT-POINTER,
```

```
      <anytype>-Picture BY-VARIANT-POINTER ).

Remove all objects in a collection.
NO-RETURN-VALUE <com-handle>: Clear (  ).

Returns the number of objects in a collection.
roperty Get:    [ Integer-Var = ] <com-handle>:Count.
Property Set:    <com-handle>:Count [ = Integer-Var ].

Returns a specific member of a Collection object (IImage) either by position or by key.
Property Get:    [ Com-Handle-Var = ] <com-handle>:Item (
          <anytype>-Index BY-VARIANT-POINTER ).

Remove a specific member of a collection.
NO-RETURN-VALUE <com-handle>: Remove (
      <anytype>-Index BY-VARIANT-POINTER ).




IImage (ListImage)
A bitmap or icon of any size that can be used in others controls.

Draws the image to a given device context (DC) at a specified location using a specified style.
imlNormal = 0
imlTransparent = 1
imlSelected = 2
imlFocus = 3
NO-RETURN-VALUE <com-handle>: Draw (
      Integer-hDC,
      <anytype>-x BY-VARIANT-POINTER,
      <anytype>-y BY-VARIANT-POINTER,
      <anytype>-Style BY-VARIANT-POINTER ).

Create an icon from a ListImage object in an ImageList control.
[ <user-defined>-Var = ] <com-handle>: ExtractIcon (  ).

Returns/sets the index of an object in a collection Read only at runtime.
Property Get:    [ Integer-Var = ] <com-handle>:Index.
Property Set:    <com-handle>:Index [ = Integer-Var ].

Retuns/sets the unique string of an object in a collection.
Property Get:    [ Character-Var = ] <com-handle>:Key.
Property Set:    <com-handle>:Key [ = Character-Var ].

Returns/sets the image Picture
Property Get:    [ <user-defined>-Var = ] <com-handle>:Picture.

Store any extra data needed for your program.
Property Get:    [ <anytype>-Var = ] <com-handle>:Tag.
Property Set:    <com-handle>:Tag [ = <anytype>-Var ].
```

Other usefull information about ImageList OCX I found at:
http://www.gritengine.com/maxscript_html/imagelist_activex_control.htm
https://www.freetutes.com/learn-vb6-advanced/lesson4/p3.html

https://knowledge.autodesk.com/search-result/caas/CloudHelp/cloudhelp/2017/ENU/MAXScript-Help/files/GUID-B2761D73-7B3D-4A71-99E1-637C34059BBB-htm.html

"The ImageList is a control that enables you to store graphic images in an application. Other controls can then use the ImageList as a central repository for the images that they will use. Both bitmaps (**.bmp files) and icons (**.ico files) can be stored in the ImageList control. At runtime the ImageList is invisible, just like a Timer or a CommonDialog control, so you can place it anywhere on a form without interfering with the user interface."

The ImageList is not a visual control (widget), it manage a central repository for Images that can be used by other controls like ListView or TreeView.
The ImageList API is exposed by IImageList COM interface. Two other COM interfaces are exposed as com-handle, IImages as ListImages respectiv IImage as ListImage.
Because ImageList is hidden at runtime, no events are fired for this control.

I found two 4gl procedures inside VM which use a ImageList for a TreeView. ImageList is the ActiveX

from dyntreeview.w

```
/* Definitions of handles for OCX Containers                            */
DEFINE VARIABLE hImageList AS WIDGET-HANDLE NO-UNDO.
DEFINE VARIABLE chhImageList AS COMPONENT-HANDLE NO-UNDO.

CREATE CONTROL-FRAME hImageList ASSIGN
       FRAME           = FRAME frTreeView:HANDLE
       ROW             = 1
       COLUMN          = 1
       HEIGHT          = 1.91
       WIDTH           = 8
       HIDDEN          = yes
       SENSITIVE       = yes.

...
IF VALID-HANDLE(chhImageList) THEN
     chhImageList:ImageList:ListImages:CLEAR().
```

from treeview.p

```
FUNCTION loadImage RETURNS INTEGER
  ( INPUT pcImageFile AS CHARACTER ) :
/*------------------------------------------------------------------------------
 Purpose:  Takes the file referenced in pcImageFile, checks to see if it is already in
           the ImageList and if not, loads it. If this is successful then its index in
           the ImageList is returned. If the file is already in the ImageList, its index is returned.

           If the load fails, 0 is returned.

 Note:    The filename must specify the relative path and filename with the extension.

          Since the imagelist does not contain any images, the ImageList property is assigned
          to the Treeview only once, after an image is added.
------------------------------------------------------------------------------*/
  DEFINE VARIABLE chImage      AS COM-HANDLE NO-UNDO.
  DEFINE VARIABLE chNewImage   AS COM-HANDLE NO-UNDO.
  DEFINE VARIABLE chTreeview   AS COM-HANDLE NO-UNDO.
  DEFINE VARIABLE hTreeview    AS HANDLE     NO-UNDO.
  DEFINE VARIABLE iImageIndex  AS INTEGER    NO-UNDO.
  DEFINE VARIABLE chImageList  AS COM-HANDLE NO-UNDO.
  DEFINE VARIABLE hImageList   AS HANDLE     NO-UNDO.
  DEFINE VARIABLE cExtension   AS CHARACTER  NO-UNDO.

  /* Check if we have an image to load */
  IF pcImageFile = ? OR
     pcImageFile = "":U THEN
    RETURN 0.
  /* Find out if the image has already been loaded. If it has we need to
   * find its index. */
  /* Get the handle to the TreeView itself. */
  {get TVCtrlFrame hTreeview }.
  ASSIGN chTreeview   = hTreeview:COM-HANDLE
         chTreeview   = chTreeview:CONTROLS:ITEM(1)
         NO-ERROR.

  {get ILCtrlFrame hImageList }.
  ASSIGN chImageList   = hImageList:COM-HANDLE
         chImageList   = chImageList:CONTROLS:ITEM(1)
         NO-ERROR.

  chImage = chImageList:ListImages:ITEM(pcImageFile) NO-ERROR.

  IF NOT VALID-HANDLE(chImage) THEN
  DO:
    /* Get the full path of the image */
    FILE-INFO:FILE-NAME = pcImageFile.

    IF FILE-INFO:FULL-PATHNAME <> ? THEN
    DO:

      /* Ensure that the image type is one that is supported by the LOAD-PICTURE statement */
      ASSIGN cExtension = ENTRY(NUM-ENTRIES(pcImageFile, ".":U),pcImageFile, ".":U) NO-ERROR.
```

```
      IF LOOKUP(cExtension,"BMP,WMF,EMF,ICO,CUR,DIB":U) = 0
      OR LOOKUP(cExtension,"BMP,WMF,EMF,ICO,CUR,DIB":U) = ? THEN
          RETURN 0.

      chImage = LOAD-PICTURE(FILE-INFO:FULL-PATHNAME).
    END.

    IF VALID-HANDLE(chImage) THEN
    DO:
      ASSIGN chNewImage  = chImageList:ListImages:ADD(chImageList:ListImages:COUNT + 1, pcImageFile, chImage)
             iImageIndex = chNewImage:INDEX.
      /* assign the imagelist to the treeview */
      IF chImageList:ListImages:COUNT = 1 THEN DO:
          chTreeview:ImageList = chImageList.
      END.
    END.
  END.
  ELSE
    iImageIndex = chImage:INDEX.

  RELEASE OBJECT chImage     NO-ERROR.
  RELEASE OBJECT chNewImage  NO-ERROR.
  RELEASE OBJECT chtreeview  NO-ERROR.
  RELEASE OBJECT chImageList NO-ERROR.

  RETURN iImageIndex.

END FUNCTION.
```

IMO, at a first view, ImageList (IImageList) should be converted like any other OCX into a 4gl WrappedResource not a Widget because ImageList is always hidden.
Also after conversion, ImageList is accessed by handle and a WrappedResource can be wrapped into a handle. ListImages (IImages) and ListImage (IImage) can be implemented as ComObect wrapped in comhandle.
This is similar with the implementation of IVBDataObject and IVBDataObjectFiles COM interfaces implementation in 3858b (Progress Bar implementation)
Doing so we don't need to do any changes in the conversion rules for ListImages and ListImage because ComObject access is implemented using Java reflection.
Of course they are a lot of details to discuss like how we should represent a <user-defined>-Var type for Picture and how to implement the draw() method of an IImage.

REMARK

Component Object Model (COM) is a Microsoft technology (see https://en.wikipedia.org/wiki/Component_Object_Model)
COM is the basis for several other Microsoft technologies and frameworks.

Bellow is a C++ example which figure out how COM is used in an application to create a COM object and query an interface.

```
HRESULT hr = NULL;

// create ImageList object and get IImageList interface
hr = ::CoCreateInstance(CLSID_IImageList,
                        NULL,
                        CLSCTX_INPROC_SERVER,
                        IID_IImageList,
                        (void **) &pImageList);
if (hr != S_OK)
{
  return FALSE;
}

// get IListImages interface
hr = (*IID_IImageList) pImageList->QueryInterface(IID_IListImages,
                                                  (void **) &pListImages;
if (hr != S_OK)
{
 return FALSE;
}
```

Each COM object has a unique CLSID and each COM interface has a unique IID. Both are GUID's (UUID's).
In conclusion COM objects and interfaces are handled by their ID's, IImageList for example is just a friendly name.
That make it possible to have multiple IImageList COM objects for example each having it own unique ID but exposing different properties and methods.
This is important to know that in order to select the appropriate COM interface, in our case from mscomctl.ocx

#16 - 11/06/2019 09:07 AM - Greg Shah

At this time, please minimize the work on this to only those features needed for the application being converted. I think this means that the ListView support is postponed and we only need to support the usage for TreeView.

Hynek: Please make a list of the known work that is needed.

#17 - 11/06/2019 09:44 AM - Marius Gligor

1. Today I tried to convert some UAST tests for IMAGELIST OCX component and I found an expression like:

```
chControlFrame:ImageList:ListImages:CLEAR().
```

Here chControlFrame:ImageList should be converted as handle, ListImages is a comhandle attribute and CLEAR() is a method.
Unfortunately I cannot covert the expression I've got a warning mesage:
WARN: FWD OCX Extension: Failed to resolve legacy name for ListImages:CLEAR. Is ListImages:CLEAR defined in interface com.goldencode.p2j.ui.ImageList?
and the generated Java code is:

```
NoopCom.handle().chain("ImageList").chain("ListImages").call("CLEAR");
```

which basically is a no operation expression. Next I split the expression as follow:

```
DEFINE VARIABLE hImgList AS COMPONENT-HANDLE NO-UNDO.

hImgList = chControlFrame:ImageList:ListImages.
hImgList:CLEAR().
```

which generate a Java code like:

```
comhandle hImgList = TypeFactory.comhandle();
hImgList.assign(hImageList.unwrapImageList().getListImages());
hImgList.call("clear");
```

The generated code is now OK and the clear() method is called.

I found many such chained expressions, having the same conversion issue for example:

```
ASSIGN chNewImage  = chControlFrame:ImageList:ListImages:ADD(chControlFrame:ListImages:COUNT + 1, pcImageFile,
 chImage)
       iImageIndex = chNewImage:INDEX.
```

which is not converted well,

```
chNewImage.assign(NoopCom.handle().chain("ImageList").chain("ListImages").call("ADD", ComParameter.input(Dynam
icOps.plus(hImageList.chain("ListImages").getProperty("COUNT"), new integer(1))),
```

but doing a small change, the same split:

```
DEFINE VARIABLE hImgList AS COM-HANDLE NO-UNDO.

hImgList = chControlFrame:ImageList:ListImages.

ASSIGN chNewImage  = hImgList:ADD(hImgList:COUNT + 1, pcImageFile, chImage)
       iImageIndex = chNewImage:INDEX.
```

this code is converted well. Generated Java code:

```
hImgList.assign(hImageList.unwrapImageList().getListImages());
chNewImage.assign(hImgList.call("ADD", ComParameter.input(DynamicOps.plus(hImgList.getProperty("COUNT"), new i
nteger(1))), ComParameter.input(pcImageFile), ComParameter.input(chImage)));
iImageIndex.assign(chNewImage.getProperty("INDEX"));
```

The question is, should we add a new rule to evaluate this kind of expression or we can change the code as I did in second example?

2. Regarding to minimize the implementation I have a small question. Should we implement an OCX compliant version or a custom one as I found in the initial version?

#18 - 11/07/2019 03:54 AM - Hynek Cihlar

Marius Gligor wrote:

> The question is, should we add a new rule to evaluate this kind of expression or we can change the code as I did in second example?

The method/prop chaining is something not fully implemented. When I was crafting the ocx rules I didn't encounter these. The ocx rules must be improved so that the chained expressions are properly converted. We don't want to end up with the high level object converted into a widget while the inner objects converted to com calls. The place that handles resolution of legacy methods/props is in annotations/ocx_conversion.rules, look for "get the prop/method name, consider chaining".

> 2. Regarding to minimize the implementation I have a small question. Should we implement an OCX compliant version or a custom one as I found in the initial version?

The idea was to have the imagelist ocx converted as a regular legacy widget/resource, so that it can become part of the API of the other widgets. Like the treeview widget.

#19 - 11/07/2019 04:12 AM - Hynek Cihlar

Hynek Cihlar wrote:

> The idea was to have the imagelist ocx converted as a regular legacy widget/resource, so that it can become part of the API of the other widgets. Like the treeview widget.

Marius, I reread your note #3856-15 and I think what you propose makes sense. IImageList and IImage do behave as simple containers and so I don't see any reason why not to expose them as com-handles. This will simplify the conversion and the overall implementation greatly. So to fix the conversion issue above ocx_conversion.rules will have to be modified:

```
                  <rule>name1 == null
                     <rule>[methProp is not of type com-handle]
                     <action>
                        printfln("WARN: FWD OCX Extension: Failed to resolve legacy name for %s. Is %s defined
in %s?", name, name, widgetClass)
                     </action>

                     <action>copy.putAnnotation("noop-com", true)</action>
                     <action>rootComInvocCpy = null</action>
                     <action>wHandleRef.remove()</action>
                     </rule>
                  </rule>
```

You will have to come up with the correct interpretation of "methProp is not of type com-handle" above.

#20 - 11/07/2019 09:16 AM - Marius Gligor

1. Looking over initial ImageList OCX replacement implementation I found:
- ImageListWidget is implemented as a widget which extends BaseEntity<BaseConfig> having 2 methods defined:

```
  /**
    * Adds an image to the list.
    *
    * @param   path
    *          The image file path. Can be an absolute name or relative. When relative the Propath entries
    *          will be searched.
    *
    * @return  the unique id of the created image or unknown.
    */
   @LegacyMethod(name = "ADD-IMAGE")
   integer addImage(character path);

   /**
    * Returns all image ids stored in the list.
    *
    * @return  see above.
    */
   integer[] getImageIds();
```

and 1 overrided to clean-up the global images:

```
   @Override
   protected boolean resourceDelete()
   {
      boolean delete = super.resourceDelete();
      if (delete)
      {
         LogicalTerminal.getClient().removeGlobalImages(getId());
      }

      return delete;
   }
```

- only addImage is a legacy method.
- in TreeWidgetBase I found:

```
   public void setImageList(handle imgList)
   {
      if (BaseDataType.isAllKnown(imgList))
      {
         imageList = new handle(imgList);
         config.imageListId = ((GenericWidget) imgList.getResource()).getId();
      }
      else
      {
         imageList = null;
         config.imageListId = -1;
      }

      pushWidgetAttr("imageListId", config.imageListId);
   }
```

Which is exactly what we need when the ImageList will be fully implemented. ImageList is a 4gl widget extension accessed by handle, and in the client side ImageList widget id is useful to find ImageListGuiImpl, which is the client side ImageList widget always hidden.

2. What I did so far:
I improved ImageListWidget which extends now BaseEntity<ImageListConfig> In other words, I added a specific configuration file for widget ImageListConfig
I added all methods and properties of the replaced OCX to make conversion works.

For simplicity I provided dummy values and implementation for methods and properties except for
ListImages add(integer index, character key, comhandle picture) which delegate the call to original addImage method

As I understood the picture (image) located at file path is loaded into a global collection on client side returning a unique id on success.
The key parameter is the file path in OCX implementation.

```
public comhandle add(integer index, character key, comhandle picture)
{
   ImageListWidget w = (ImageListWidget) parent;
   w.addImage(key); // delegate call
   return picture;
}
```

In the future we could provide a complete implementation of ImageList since we already have the skeleton.
I tested using all my UAST tests, by splitting the chained expressions and both conversion and execution are working fine.

Hynek, please let me know:
Is this enough to use ImageList with TreeView?
Do you think that other ImageList functionality should be added at this stage, apart of fixing the chaining expressions in conversion?

#21 - 11/07/2019 09:21 AM - Hynek Cihlar

Marius Gligor wrote:

> Hynek, please let me know:
> Is this enough to use ImageList with TreeView?

AFAIK this is enough.

> Do you think that other ImageList functionality should be added at this stage, apart of fixing the chaining expressions in conversion?

Do you refer to functionality used in a GUI app?

#22 - 11/07/2019 09:27 AM - Marius Gligor

Hynek Cihlar wrote:

> Do you refer to functionality used in a GUI app?

So far I worked only in p2j project.

#23 - 11/12/2019 08:05 AM - Marius Gligor

I'm working to define conversion rules for OCX statements which combine handle calls with com-handle calls.
So far I defined rules which are working with simple expressions.For example the following statements:

```
chControlFrame:ImageList:ListImages:count().
chControlFrame:ImageList:ListImages:item(1).
chControlFrame:ImageList:ListImages:ADD(15, "2", 99).
```

are now converted into valid Java code as follow:

```
hImageList.unwrapImageList().getListImages().call("count");
hImageList.unwrapImageList().getListImages().call("item", ComParameter.input(1));
hImageList.unwrapImageList().getListImages().call("ADD", ComParameter.input(15), ComParameter.input("2"), ComP
arameter.input(99));
```

But I found another interesting case in which a parameter or more are an expression of mixed calls like in the following wxample:

chControlFrame:ImageList:ListImages:ADD(chControlFrame:ImageList:ListImages:COUNT + 1, "2", 99).

which is converted into Java code as follow:

hImageList.unwrapImageList().getListImages().call("ADD",
ComParameter.input(DynamicOps.plus(hImageList.chain("ImageList").chain("ListImages").getProperty("COUNT"), new integer(1))),
ComParameter.input("2"), ComParameter.input(99));

Unfortunately the following generated Java code is not a valid code because hImageList ia a handle and doesn't have a method chain which is valid only for comhandle type
hImageList.chain("ImageList").chain("ListImages").getProperty("COUNT")

Having nested expressions sounds as recursion!
Can we use user functions defined in rules called as "execLib(...)" to implement a recursive algorithm?

NOTE. In all examples chControlFrame:ImageList is converted into hImageList which is the ImageList widget handle and ListImages is a widget property which return a comhandle.

#24 - 11/12/2019 09:29 AM - Marius Gligor

I did a simple test using a simple walk rule:

```
<rule> type == prog.var_com_handle
   <action>printfln("%s", this.id)</action>
</rule>
```

and I've got:

```
25769804196
25769804208
25769804277
25769804289
```

the var_com_handle node that I'm looking for is 25769804289
With the actual code in

```
<rule> type == prog.var_com_handle
   <action>printfln("%s", this.id)</action>
   .... existing rules ...
```

I've got only

```
25769804196
25769804208
25769804277
```

So, during the walk for some reason the node with id=25769804289 is not visited! I have to check why.
PS. The node with id=25769804289 is in tree.

#25 - 11/12/2019 10:04 AM - Hynek Cihlar

Marius Gligor wrote:

> Unfortunately the following generated Java code is not a valid code because hImageList ia a handle and doesn't have a method chain which is valid only for comhandle type
> hImageList.chain("ImageList").chain("ListImages").getProperty("COUNT")
>
> Having nested expressions sounds as recursion!
> Can we use user functions defined in rules called as "execLib(...)" to implement a recursive algorithm?

For some reason hImageList.chain("ImageList").chain("ListImages") isn't converted to hImageList.unwrapImageList().getListImages() in the DynamicOps.plus argument. Isn't this the issue?

#26 - 11/12/2019 10:08 AM - Hynek Cihlar

Marius Gligor wrote:

> I did a simple test using a simple walk rule:
>
> [...]
>
> and I've got:
> [...]
>
> the var_com_handle node that I'm looking for is 25769804289
> With the actual code in
> [...]
> I've got only
> [...]
>
> So, during the walk for some reason the node with id=25769804289 is not visited! I have to check why.
> PS. The node with id=25769804289 is in tree.

The root com handles are converted to widget handles, see copy.parent.putAnnotation("type", #(long) prog.var_handle) in ocx_conversion.rules. The conversion is performed on the AST copy, so you will still find the com-handle in the origin AST.

#27 - 11/12/2019 11:13 AM - Marius Gligor

Hynek Cihlar wrote:

> Marius Gligor wrote:
>
> > Unfortunately the following generated Java code is not a valid code because hImageList ia a handle and doesn't have a method chain which is valid only for comhandle type
> > hImageList.chain("ImageList").chain("ListImages").getProperty("COUNT")
> >
> > Having nested expressions sounds as recursion!
> > Can we use user functions defined in rules called as "execLib(...)" to implement a recursive algorithm?
>
> For some reason hImageList.chain("ImageList").chain("ListImages") isn't converted to hImageList.unwrapImageList().getListImages() in the DynamicOps.plus argument. Isn't this the issue?

Yes, this is the issue.

#28 - 11/12/2019 01:15 PM - Ovidiu Maxiniuc

The chain is used (or should be used) only when the first "link" (in this case hImageList) is a comhandle. Somehow the conversion assumed this is the case. Otherwise the standard handle API is used, meaning the unwrap___ method is injected. If the type of the hImageList was altered during processing (too early or too late), it is possible that the incorrect API to be emitted.

#29 - 11/12/2019 01:22 PM - Marius Gligor

Marius Gligor wrote:

> Hynek Cihlar wrote:
>
> > Marius Gligor wrote:
> >
> > > Unfortunately the following generated Java code is not a valid code because hImageList ia a handle and doesn't have a method chain which is valid only for comhandle type
> > > hImageList.chain("ImageList").chain("ListImages").getProperty("COUNT")
> > >
> > > Having nested expressions sounds as recursion!
> > > Can we use user functions defined in rules called as "execLib(...)" to implement a recursive algorithm?
> >
> > For some reason hImageList.chain("ImageList").chain("ListImages") isn't converted to hImageList.unwrapImageList().getListImages() in the DynamicOps.plus argument. Isn't this the issue?

> Yes, this is the issue.

As I understood we convert chControlFrame to handle But we have the following chain:

chControlFrame:ImageList:ListImages: ...

- chControlFrame is the OCX com-handle which is converted to widget handle.
- ImageList is a com property which seems to not be used on conversion. How we should proceed this node?. Should we ignore it?
- ListImages is a ImageList widget legacy method, and we have to resolve the legacy name and type at ast walk time.

Doing that finally the Java converted code looks good:

```
hImageList.unwrapImageList().getListImages()
```

Please let me know if I understood correctly.
I forgot to mention that chControlFrame:ImageList is defined in ext-hints file as com-handle="chControlFrame:ImageList"

#30 - 11/12/2019 01:34 PM - Marius Gligor

Ovidiu Maxiniuc wrote:

> The chain is used (or should be used) only when the first "link" (in this case hImageList) is a comhandle. Somehow the conversion assumed this is the case. Otherwise the standard handle API is used, meaning the unwrap___ method is injected. If the type of the hImageList was altered during processing (too early or too late), it is possible that the incorrect API to be emitted.

Ovidiu, hImageList should be a handle and the next token ImageList is a com property.
According to my tests unwrap___ is injected if after hImageList handle, a legacy method or attribute is used in my case ListImages. From legacy name the widget is resolved and the corresponding unwrap___ is injected.
What I don't understand is how we have to deal with ImageList node. So far I ignored that node, skip to the next node which is the legacy method ListImages, fixed the legacy in conversion and it works.

#31 - 11/12/2019 02:59 PM - Hynek Cihlar

Marius Gligor wrote:

> What I don't understand is how we have to deal with ImageList node. So far I ignored that node, skip to the next node which is the legacy method ListImages, fixed the legacy in conversion and it works.

See the comment 4. Convert chained com property from <chCF>:<com-property-name>:<meth-or-attr> to hWidget:<meth-or-attr> in ocx_conversion.rules and the rules below that comment.

#32 - 11/12/2019 03:05 PM - Hynek Cihlar

Hynek Cihlar wrote:

> Marius Gligor wrote:
>
> > What I don't understand is how we have to deal with ImageList node. So far I ignored that node, skip to the next node which is the legacy method ListImages, fixed the legacy in conversion and it works.
>
> See the comment 4. Convert chained com property from <chCF>:<com-property-name>:<meth-or-attr> to hWidget:<meth-or-attr> in ocx_conversion.rules and the rules below that comment.

And the expected ext-hints are:

```
<extension>
   <ocx com-handle="chCtrlFrame:ImageList" target-widget="imagelist" target-handle="hImageList" control-frame=
"CtrlFrame" control-frame-handle="chCtrlFrame"/>
</extension>
```

#33 - 11/13/2019 11:17 AM - Marius Gligor

For chControlFrame:ImageList:ListImages:ADD(chControlFrame:ImageList:ListImages:COUNT + 1, "2", 99). expression, doing nothing in conversion
I've got the foolowing node in .ast file.

```
<ast col="0" id="25769804272" line="0" text="assignment" type="ASSIGNMENT">
  <ast col="0" id="25769804273" line="0" text="expression" type="EXPRESSION">
    <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
    <annotation datatype="java.lang.Long" key="peerid" value="193273528530"/>
    <ast col="36" id="25769804274" line="79" text=":" type="COM_INVOCATION">
      <annotation datatype="java.lang.Long" key="peerid" value="193273528531"/>
      <ast col="25" id="25769804275" line="79" text=":" type="COM_INVOCATION">
        <annotation datatype="java.lang.Long" key="peerid" value="193273528532"/>
        <ast col="15" id="25769804276" line="79" text=":" type="COM_INVOCATION">
          <annotation datatype="java.lang.Long" key="peerid" value="193273528533"/>
          <ast col="1" id="25769804277" line="79" text="chControlFrame" type="VAR_COM_HANDLE">
            <annotation datatype="java.lang.Long" key="oldtype" value="2764"/>
            <annotation datatype="java.lang.Long" key="refid" value="25769803811"/>
            <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
            <annotation datatype="java.lang.Long" key="peerid" value="193273528534"/>
          </ast>
          <ast col="16" id="25769804278" line="79" text="ImageList" type="COM_PROPERTY">
            <annotation datatype="java.lang.Long" key="oldtype" value="2696"/>
            <annotation datatype="java.lang.Boolean" key="com-call" value="true"/>
            <annotation datatype="java.lang.Long" key="peerid" value="193273528535"/>
          </ast>
        </ast>
        <ast col="26" id="25769804279" line="79" text="ListImages" type="COM_PROPERTY">
          <annotation datatype="java.lang.Long" key="oldtype" value="2764"/>
          <annotation datatype="java.lang.Boolean" key="com-call" value="true"/>
          <annotation datatype="java.lang.Long" key="peerid" value="193273528536"/>
        </ast>
      </ast>
      <ast col="37" id="25769804280" line="79" text="ADD" type="COM_METHOD">
        <annotation datatype="java.lang.Long" key="oldtype" value="500"/>
        <annotation datatype="java.lang.Boolean" key="com-call" value="true"/>
        <annotation datatype="java.lang.Long" key="peerid" value="193273528537"/>
        <ast col="0" id="25769804282" line="0" text="" type="COM_PARAMETER">
          <annotation datatype="java.lang.Long" key="peerid" value="193273528538"/>
          <ast col="83" id="25769804283" line="79" text="+" type="PLUS">
            <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
            <annotation datatype="java.lang.Boolean" key="dynamic" value="true"/>
            <annotation datatype="java.lang.Long" key="peerid" value="193273528539"/>
            <ast col="76" id="25769804286" line="79" text=":" type="COM_INVOCATION">
              <annotation datatype="java.lang.Long" key="peerid" value="193273528540"/>
              <ast col="65" id="25769804287" line="79" text=":" type="COM_INVOCATION">
                <annotation datatype="java.lang.Long" key="peerid" value="193273528541"/>
                <ast col="55" id="25769804288" line="79" text=":" type="COM_INVOCATION">
                  <annotation datatype="java.lang.Long" key="peerid" value="193273528542"/>
                  <ast col="41" id="25769804289" line="79" text="chControlFrame" type="VAR_COM_HANDLE">
                    <annotation datatype="java.lang.Long" key="oldtype" value="2764"/>
                    <annotation datatype="java.lang.Long" key="refid" value="25769803811"/>
                    <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
                    <annotation datatype="java.lang.Long" key="peerid" value="193273528543"/>
                  </ast>
                  <ast col="56" id="25769804290" line="79" text="ImageList" type="COM_PROPERTY">
                    <annotation datatype="java.lang.Long" key="oldtype" value="2696"/>
                    <annotation datatype="java.lang.Boolean" key="com-call" value="true"/>
                    <annotation datatype="java.lang.Long" key="peerid" value="193273528544"/>
                  </ast>
                </ast>
                <ast col="66" id="25769804291" line="79" text="ListImages" type="COM_PROPERTY">
                  <annotation datatype="java.lang.Long" key="oldtype" value="2764"/>
                  <annotation datatype="java.lang.Boolean" key="com-call" value="true"/>
                  <annotation datatype="java.lang.Long" key="peerid" value="193273528545"/>
                </ast>
              </ast>
              <ast col="77" id="25769804292" line="79" text="COUNT" type="COM_PROPERTY">
                <annotation datatype="java.lang.Long" key="oldtype" value="1165"/>
                <annotation datatype="java.lang.Boolean" key="com-call" value="true"/>
                <annotation datatype="java.lang.Long" key="peerid" value="193273528546"/>
              </ast>
            </ast>
            <ast col="85" id="25769804293" line="79" text="1" type="NUM_LITERAL">
              <annotation datatype="java.lang.Boolean" key="is-literal" value="true"/>
              <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
```

```
                <annotation datatype="java.lang.Boolean" key="use64bit" value="false"/>
                <annotation datatype="java.lang.Long" key="peerid" value="193273528548"/>
              </ast>
            </ast>
          </ast>
          <ast col="0" id="25769804296" line="0" text="" type="COM_PARAMETER">
            <annotation datatype="java.lang.Long" key="peerid" value="193273528549"/>
            <ast col="88" id="25769804297" line="79" text="&amp;quot;2&amp;quot;" type="STRING">
              <annotation datatype="java.lang.Boolean" key="is-literal" value="true"/>
              <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
              <annotation datatype="java.lang.Long" key="peerid" value="193273528550"/>
            </ast>
          </ast>
          <ast col="0" id="25769804300" line="0" text="" type="COM_PARAMETER">
            <annotation datatype="java.lang.Long" key="peerid" value="193273528551"/>
            <ast col="93" id="25769804301" line="79" text="99" type="NUM_LITERAL">
              <annotation datatype="java.lang.Boolean" key="is-literal" value="true"/>
              <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
              <annotation datatype="java.lang.Boolean" key="use64bit" value="false"/>
              <annotation datatype="java.lang.Long" key="peerid" value="193273528552"/>
            </ast>
          </ast>
        </ast>
      </ast>
    </ast>
  </ast>
```

This node has 2 VAR_COM_HANDLE nodes. When we walk over the tree the first COM-HANDLE id="25769804277" is visited.
Doing changes to convert this COM-HANDLE the next handle id="25769804289" is never visited!.
What I'm trying to do is to navigate over this node, collect all COM-VAR_COM_HANDLE nodes and try to perform conversion for each one.
For some reasons, so far I cannot find the second VAR_COM_HANDLE in this node!

I have some questions:
1. The output of conversion is in the .ast file. As I understood we have 2 trees, main and copy. The output .ast file is a copy tree data. During conversion both main and copy trees are changed and keep in sync? Is .ast file the input and after conversion the output as well?

#34 - 11/13/2019 11:35 AM - Hynek Cihlar

Marius Gligor wrote:

> 1. The output of conversion is in the .ast file. As I understood we have 2 trees, main and copy. The output .ast file is a copy tree data. During conversion both main and copy trees are changed and keep in sync? Is .ast file the input and after conversion the output as well?

copy is the same only before rules execution starts, main and copy trees are not kept in sync. Only copy is supposed to be changed during rules execution AFAIK.

Btw, what does your ext-hints file look like?

#35 - 11/13/2019 11:43 AM - Marius Gligor

Hynek Cihlar wrote:

> Marius Gligor wrote:
>
> > 1. The output of conversion is in the .ast file. As I understood we have 2 trees, main and copy. The output .ast file is a copy tree data. During conversion both main and copy trees are changed and keep in sync? Is .ast file the input and after conversion the output as well?
>
> copy is the same only before rules execution starts, main and copy trees are not kept in sync. Only copy is supposed to be changed during rules execution AFAIK.
>
> Btw, what does your ext-hints file look like?

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<extension>
   <ocx control-frame="ControlFrame" control-frame-handle="chControlFrame" com-handle="chControlFrame:ImageLis
t" target-handle="hImageList" target-widget="imagelist" />
   <drop-procedure name="control_load" />
</extension>
```

All other conversions works fine, except nested com-handle as I already mentioned in note 23.

#36 - 11/13/2019 12:25 PM - Greg Shah

> Is .ast file the input and after conversion the output as well?

Yes. It is read as the input to create the AST tree in memory. Then the TRPL rules operate on the in memory copies, where each step in the process the copy tree is the output of one step and the input to the next step, but as the input it is named this.

When a ruleset calls persist() in post rules, it will save off the copy tree to the .ast file.

#37 - 11/14/2019 10:30 AM - Marius Gligor

Unfortunately, after many attempts, I concluded that using existing conversion rules from section 4 in ocx_conversion.rules file, it's not possible to fix conversion for all possible cases.
However, IMO I found a better approach, I cut off the existing rules in section 4 and I replaced them with other rules less complex and more generic.

What I did:
Let take a look over the expression chControlFrame:ImageList:ListImages:count().

- Tokens chControlFrame:ImageList which forms the OCX comhandle, are converted into a widget handle hImageList, according to the definition from .ext-hints file.
- Next token ImageList MUST be allways a widget legacy property or method since after conversion it is linked to a handle.
A COM interface expose methods and attributes which returns either another COM interface (comhandle) or other types except handle which obviously is not part of COM specifications. When a comhandle is returnated, a chain is possible to follow after.
- Next token count() is a com method. This chain is possible because ListImages property retuns a comhandle.

Looking at the AST tree from #note-33 I observed that doing few changes it is possible to convert the com-handle to handle in place, directly in the copy tree.
For each SAX parser event of type type == prog.var_com_handle (section 4 in ocx_conversion.rules file) the following rules and actions are applied:

- Check if chControlFrame:ImageList exists, if yes convert, chControlFrame node into a handle node (hImageList) and remove ImageList node.
- Rearrange the tree by replacing com_invocation type to colon type in parent.parent node, move handle node under this note and remove old parent node.
- Fetch next node in the chain if exists, in our case ListImages wich must be resolved as a legacy node.
- Check if the legacy node has parameters of type prog.com_parameter and if found, replace them with prog.parameter

That's all!
With the new rules in place I tested some expressions like:

```
chControlFrame:ImageList:ListImages.
chControlFrame:ImageList:ListImages:count().
chControlFrame:ImageList:ListImages:item(1).
chControlFrame:ImageList:ListImages:ADD(15, "2", 99).
chControlFrame:ImageList:ListImages:ADD(chControlFrame:ImageList:ListImages:COUNT + 1, "2", 99).
```

which were successfuly converted into Java valid code:

```
hImageList.unwrapImageList().getListImages();
hImageList.unwrapImageList().getListImages().call("count");
hImageList.unwrapImageList().getListImages().call("item", ComParameter.input(1));
hImageList.unwrapImageList().getListImages().call("ADD", ComParameter.input(15), ComParameter.input("2"), ComP
arameter.input(99));
hImageList.unwrapImageList().getListImages().call("ADD", ComParameter.input(DynamicOps.plus(hImageList.unwrapI
mageList().getListImages().getProperty("COUNT"), new integer(1))), ComParameter.input("2"), ComParameter.input
(99));
```

Expressions containing nested com-handles should now work at any nested depth.
Nevertheless I have to do more tests in order to validate the new solution.

changes has been comitted in branch 3856a rev 11336

#38 - 11/15/2019 12:39 PM - Marius Gligor

Doing tests for ImageList OCX widget conversion I found another interesting case:

```
DEFINE VARIABLE chControlFrame AS COM-HANDLE NO-UNDO.
DEFINE VARIABLE ControlFrame   AS HANDLE     NO-UNDO.

ASSIGN chControlFrame = ControlFrame:COM-HANDLE
       chControlFrame = chControlFrame:CONTROLS:ITEM(1)
       NO-ERROR.

IF VALID-HANDLE(chControlFrame) THEN
  chControlFrame:ImageWidth = piImageWidth NO-ERROR.
```

Searching in the ABL documentation I found that OCX com-handle can be obtained in 2 ways.
https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dvpin/accessing-the-control-handle.html
However in FWD we always convert control frame into widget handle and such expressions makes no sense.
So I added rules to neutralize these kind of expressions which are converted in Java as follow:

```
silent(() ->
{
  hImageList.assign(hImageList);
  hImageList.assign(hImageList);
});
```

Basically the converted Java code is equivalent with a NOP (no operation).
Another option is to ignore and remove those expressions, but I think that is better to keep them even thought they do nothing.
As a result of testing I improved and optimized the conversion rules.
All OCX expressions I tested so far were converted successfully. In the future if w'll find other conversion cases we could fix them at that time.

I consider that changes for this task are done and branch 3856a is ready for a code review.

#39 - 11/19/2019 12:03 PM - Marius Gligor

In order to avoid any problems merging branch 3858b into branch 3856a finally I did:

1. I created a new branch 3856b from current trunk.
2. I merged branch 3856a into new branch 3856b. Than I fixed all merge conflicts.
All uast tests for ImageList widget were successfully converted using the code from new branch.
3. Next I merged the changes from branch 3858b branch, related to ProgressBar widget implementation,
into the new branch 3856b. Than I fixed all merge conflicts.
All uast tests for ImageList widget were successfully converted using the code from new branch.
Unfortunately not all uast tests for ProgressBar widget were successfully converted using the code from new branch.
ocx_conversion.rules file looks slightly different in branch 3858b compared to branch 3856a
4. Next I'm going to check why not all ProgressBar widgets are not converted and fix the issue.

#40 - 11/20/2019 08:06 AM - Marius Gligor

I've fixed OCX conversion rules, then I tested both progress-bar and imagelist uast tests and all were successfully converted.
Also I tested the runtime behavior of converted tests and I found that are working properly.
Uast tests are located in testcases project, uast/progressbar/ and uast/image-list/ folders.
All changes are in branch 3856b ready for code review.

#41 - 11/20/2019 09:49 AM - Marius Gligor

Documentation for PROGRESS-BAR and IMAGELIST widgets, could help for code review:
https://proj.goldencode.com/projects/p2j/wiki/Progress_Indicator_Widget_and_OCX_Replacement
https://proj.goldencode.com/projects/p2j/wiki/ImageList_Widget_and_OCX_Replacement

#42 - 11/20/2019 01:51 PM - Greg Shah

- Status changed from WIP to Review

#43 - 12/17/2019 06:30 PM - Greg Shah

I've confirmed with Hynek that he is good with all the final changes in 3856b. The next step is to get it tested so that it can be merged to trunk.

- conversion testing in both large GUI applications
- GUI runtime testing in both large GUI applications and in Hotel GUI

#44 - 12/19/2019 01:45 PM - Hynek Cihlar

Code review 3856b revision 11357. These are the merged changes for TAB-SET widget from 3767b.

TabSetWidget.show()
I think there should be a call to _setVisible(true). Even better, just call GenericWidget.view(), which does exactly this.

TabSetGuiImpl
TAB_HEIGHT_EXTRA_SPACE and TAB_WIDTH_EXTRA_SPACE miss javadocs.
The draw logic in draw() should be in the theme classes.

Windows10Theme please remove the added white spaces.

#45 - 12/20/2019 03:31 AM - Hynek Cihlar

On 12/20/19 9:23 AM, Marius Gligor wrote:

> Hynek,
>
> OK i'll do that and let you know when I'm finished and ready to review my changes.
>
> Thanks,
> Marius
>
> On 20.12.2019 10:20, Hynek Cihlar wrote:
>
> > Marius,
> >
> > yes, please implement the proposed change.
> >
> > Thanks,
> > Hynek
> >
> > On 12/20/19 9:19 AM, Marius Gligor wrote:
> >
> > > Hynek,
> > >
> > > This is what I found initially in the conversion rules. We can improve the rules as you mentioned before as follow:
> > >
> > > <rule>reftype1 == -1
> > > <action>copy.putAnnotation("noop-com", true)</action>
> > > <rule>
> > >
> > > I this OK? I can do that just now because I'm already work to do changes you suggested on the code review feedback
> > >
> > > Thanks,
> > > Marius
> > >
> > > On 20.12.2019 10:12, Hynek Cihlar wrote:
> > >
> > > > Marius,
> > > >
> > > > I think the noop-com annotation should be added if either of the method name or the keyword token is not resolved. I wonder how this worked in trunk (or 3809e). I guess the assignment was removed altogether? The NoopComHandle should be generated for any props/methods that we can't convert. This tags all the places that need additional attention to be resolved. Ideally there should be no NoopComHandle references in the converted code and no skipped com methods or properties.
> > > >
> > > > Thanks,
> > > > Hynek
> > > >
> > > > On 12/20/19 9:04 AM, Marius Gligor wrote:
> > > >
> > > > > Hi Hynek,
> > > > >
> > > > > Legacy name is found name1 = SET-NODE-BGCOLOR but prog.lookupKeywordToken(name1) return -1.
> > > > > In ocx_conversion rules a noop-com annotation is added only if the legacy name is not found.
> > > > >
> > > > > <rule>name1 == null
> > > > > <action>printfln("WARN: FWD OCX Extension: Failed to resolve legacy name for %s. Is %s defined in %s?", name, name, widgetClass)</action>
> > > > > <action>copy.putAnnotation("noop-com", true)</action>
> > > > > </rule>
> > > > >
> > > > > If legacy name is not null the conversion rule is:
> > > > >
> > > > > <rule>name1 != null
> > > > > <action>reftype = prog.lookupKeywordToken(name1)</action>
> > > > > <rule>reftype == -1
> > > > > <action>printfln("WARN: FWD OCX Extension: Failed to resolve keyword for %s.", name1)</action>
> > > > > <action on="false">reftype1 = lookupAttributeOrMethod(reftype)</action>
> > > > > <rule on="false">reftype1 == -1
> > > > > <action>printfln("WARN: FWD OCX Extension: Failed to resolve attribute or method type for %s.", name1)</action>
> > > > > </rule>
> > > > > </rule>
> > > > >
> > > > > <rule>reftype1 != -1
> > > > > <!-- set legacy node properties -->
> > > > > <action>ref2.type = reftype1</action>

<action>ref2.text = name1</action>
<action>ref2.putAnnotation("oldtype", #(long)reftype)</action>

<!-- lookup for com parameters and replace with parameters -->
<action>ref1 = ref2.firstChild</action>
<while>ref1 != null
<action>ref2 = null</action>
<rule>ref1.type == prog.com_parameter
<action>ref1.type = prog.parameter</action>
<action>ref2 = ref1</action>
</rule>
<action>ref1 = ref1.nextSibling</action>

<!-- check for empty parameter and remove it -->
<rule>ref2 != null and ref2.numChildren 0
<action>ref2.remove()</action>
</rule>
</while>
</rule>
</rule>

When reftype1 -1 we don't touch the node otherwise we set the correct values.
I think this is OK, is it?

Thanks,
Marius

On 20.12.2019 09:47, Hynek Cihlar wrote:

> Eric,
>
> thanks for the note. I'm aware of that, that is why I want to compare the behavior with trunk. AFAIK this case should be handled there.
>
> Thanks,
> Hynek
>
> On 12/20/19 8:45 AM, Eric Faulhaber wrote:
>
>> Hynek,
>>
>> Note that Marius is not on 3809e. He's on a custom branch for his work.
>>
>> Thanks,
>> Eric
>>
>> On 12/20/19 2:42 AM, Hynek Cihlar wrote:
>>
>>> Marius,
>>>
>>> I see two issues. First, the SET-NODE-FGCOLOR and GET-NODE-FGCOLOR keywords are not defined. I can take care of that. Second, in such case, when the legacy keyword is not found, the NoopComHandle expression should be emitted. How does it convert in trunk, can you compare?
>>>
>>> Thanks,
>>> Hynek
>>>
>>> On 12/20/19 7:39 AM, Marius Gligor wrote:
>>>
>>>> Greg,
>>>>
>>>> Customer_GUI_App full conversion ended after 14 hour and 24 minutes with an conversion error.
>>>> According to conversion log messages the error occurred converting procedure:
>>>> ./abl/Customer_GUI_AppOpenEdge/customer_tree.w
>>>> As understood the problem is in the assignment for id = 72816875554764:
>>>>
>>>> chCtrlFrame:vmacmtree1:NodeFgColor(p-ptr) = color-table:get-rgb-value(...).
>>>>
>>>> in the statement:
>>>>
>>>> if ... then
>>>> assign chCtrlFrame:vmacmtree1:NodeBgColor(p-ptr) = color-table:get-rgb-value(...)
>>>> chCtrlFrame:vmacmtree1:NodeFgColor(p-ptr) = color-table:get-rgb-value(...).
>>>>
>>>> I extracted the ast tree corresponding to this statement from customer_tree.ast.

At a first view I don't see any problem. NodeFgColor is similar with NodeBgColor but only NodeFgColor raise conversion error!
The only difference i found is the annotation for oldtype:

<ast col="35" id="72816875554747" line="3666" text="SET-NODE-BGCOLOR" type="METH_VOID">
<annotation datatype="java.lang.Long" key="oldtype" value="2673"/>

<ast col="35" id="72816875554768" line="3667" text="SET-NODE-FGCOLOR" type="METH_VOID">
<annotation datatype="java.lang.Long" key="oldtype" value="-1"/>

which is 2673 for SET-NODE-BGCOLOR but -1 for SET-NODE-FGCOLOR

Searching for 2673 I found KW_SET_N_BG=2673; KW_SET_N_BG is defined in progress.g as follow
new Keyword("set-node-bgcolor" , 0, KW_SET_N_BG, false), // FWD extension, not a real 4GL keyword

But for "set-node-fgcolor" no keyword is defined in progress.g that's why oldtye = -1!
Searching the entire project for "set-node-fgcolor":

$ grep -ri "set-node-fgcolor" ./
./src/com/goldencode/p2j/ui/TreeFace.java: @LegacyMethod(name = "SET-NODE-FGCOLOR")
Binary file ./build/classes/com/goldencode/p2j/ui/TreeFace.class matches
Binary file ./build/classes.aop/com/goldencode/p2j/ui/TreeFace.class matches

I found only Legacy method annotation which means the conversion for SET-NODE-FGCOLOR is not yet implemented complete, only legacy name is defined!
Hynek, can you check please.

I attached, conversion log file, customer_tree.* files and Customer_GUI_App-error.txt file on which I extracted ABL code, AST tree and conversion error message.

Thanks,
Marius

#46 - 12/20/2019 06:43 AM - Marius Gligor

Hynek Cihlar wrote:

> Code review 3856b revision 11357. These are the merged changes for TAB-SET widget from 3767b.
>
> TabSetWidget.show()
> I think there should be a call to _setVisible(true). Even better, just call GenericWidget.view(), which does exactly this.
>
> TabSetGuiImpl
> TAB_HEIGHT_EXTRA_SPACE and TAB_WIDTH_EXTRA_SPACE miss javadocs.
> The draw logic in draw() should be in the theme classes.
>
> Windows10Theme please remove the added white spaces.

- I did the changes requested in your code review. Also I did other small code improvements.
- For OCX conversion I added rules to put "noop-com" annotation when keyword cannot be resolved and when attribute or method type cannot be resolved as well.

Please do a review over my changes in branch 3856b rev 11358, 11359 and 11360

#47 - 12/23/2019 05:38 AM - Hynek Cihlar

Code review 3856b rev 11358, 11359 and 11360.

The changes are good. Please note that I removed added white spaces from Windows10Theme.java, this was the only change in the file.

#48 - 01/07/2020 10:20 AM - Greg Shah

During conversion regression testing with a large customer application, Marius found a regression that converts without abend but manifests as a compile failure because the Java code is not correct.

The problem is occurring with indexed COM properties which are also METH_* values in the corresponding new "built-in" 4GL widget. For example:

```
ctrl-frame-handle:tree-ocx:NodeBgColor(idx) = color-table:get-rgb-value(clr-idx).
```

will be generated as:

```
treeOcx.unwrapTreeFace().setNodeBgColor(idx);
new void(ColorTable.getRgbValue(clrIdx));
```

Questions:

- Should we be treating this as an OCX/COM conversion here or as a 4GL method/attribute?
- What should the resulting Java code look like?
- Does this problem only exist after 3856b? If so, is it caused by the improvements in COM/OCX conversion that are in this branch?

#49 - 01/07/2020 11:09 AM - Hynek Cihlar

Greg Shah wrote:

- Does this problem only exist after 3856b? If so, is it caused by the improvements in COM/OCX conversion that are in this branch?

Indexed OCX properties convert OK in trunk, this must be a regression.

#50 - 01/07/2020 11:43 AM - Marius Gligor

Hynek Cihlar wrote:

> Greg Shah wrote:
>
> - Does this problem only exist after 3856b? If so, is it caused by the improvements in COM/OCX conversion that are in this branch?
>
> Indexed OCX properties convert OK in trunk, this must be a regression.

Indeed, I checked the trunk ocx_conversion.rules and I found special rules for indexed properties. Those rules are missing in 3856b and I have to fix that.

```
<!-- is this an indexed COM property being assigned? -->
<rule>rootComInvoc.parent.type == prog.assign and
      rootComInvoc.indexPos == 0 and
      methProp.getImmediateChild(prog.com_parameter, null) != null

   <!-- get the assigning expression and make it a parameter -->
   <action>ref1 = rootComInvocCpy.nextSibling</action>
   <!-- the com parameter will be converted below -->
   <action>ref1.type = prog.com_parameter</action>
   <action>ref1.text = ""</action>
   <action>ref1.move(methPropCpy, -1)</action>

</rule>
```

#51 - 01/08/2020 06:39 AM - Marius Gligor

The idea behind conversion rules for indexed OCX properties is to transform the assignment right-side value into a COM property:
chCtrlFrame:tree1:NodeBgColor(1) = COLOR-TABLE:GET-RGB-VALUE(15).
is transformed into:
chCtrlFrame:tree1:NodeBgColor(1, COLOR-TABLE:GET-RGB-VALUE(15)).
As a result, in Java the setNodeBgColor method signature is void setNodeBgColor(NumberType nodeId, NumberType bgrValue) which reflect this transformation.

In order to fix the indexed OCX properties in my branch 3856b I created a very simple UAST test:

```
DEFINE VARIABLE lColors AS LOGICAL NO-UNDO.
DEFINE VARIABLE CtrlFrame AS WIDGET-HANDLE NO-UNDO.
DEFINE VARIABLE chCtrlFrame AS COM-HANDLE NO-UNDO.

IF lColors THEN
    ASSIGN chCtrlFrame:tree1:NodeBgColor(1) = COLOR-TABLE:GET-RGB-VALUE(15)
           chCtrlFrame:tree1:NodeFgColor(2) = COLOR-TABLE:GET-RGB-VALUE(1).
```

First I converted this test using the current trunk code and I obtained:

```
 if ((lColors).booleanValue())
 {
    hTree.unwrapTreeFace().setNodeBgColor(new integer(1), ColorTable.getRgbValue(15));
    hTree.assign(ColorTable.getRgbValue(1));
 }
```

As we can see the NodeBgColor property is converted as we expect but NodeFgColor property conversion is broken due to the partial implementation.
Next I started a new conversion, this time using the code from 3856b and basically I reproduced the same bug we have in the large project conversion:

```
 if ((lColors).booleanValue())
 {
    hTree.unwrapTreeFace().setNodeBgColor(new integer(1);
    new void(ColorTable.getRgbValue(15));
    NoopCom.handle().chain("tree1").setIndexedProperty("NodeFgColor", ColorTable.getRgbValue(1), ComParameter.
input(2));
 }
```

Then I added the missing conversion rules for indexed OCX properties in 3856b branch, ocx_conversion.rules file.
When I did a new conversion using the fixed code from my branch 3856b I've got:

```
 if ((lColors).booleanValue())
 {
    hTree.unwrapTreeFace().setNodeBgColor(new integer(1), ColorTable.getRgbValue(15));
    NoopCom.handle().chain("tree1").setIndexedProperty("NodeFgColor", ColorTable.getRgbValue(1), ComParameter.
input(2));
 }
```

Now NodeBgColor property is converted as we expect and NodeFgColor partial implementation issue is fixed as a NoopCom chain.
Also I did a conversion for a list of uast tests (progressbar, tabset and my treeview) and I found that conversion works properly for all tests.

My changes has been committed in branch 3856b rev 11362.
Hynek, please do a code review on my changes, then I could restart the large project conversion.

#52 - 01/08/2020 08:51 AM - Hynek Cihlar

Code review 3856b revision 11362. The changes are good. Also please test referencing indexed properties.

#53 - 01/08/2020 08:55 AM - Marius Gligor

Hynek Cihlar wrote:

> Code review 3856b revision 11362. The changes are good. Also please test referencing indexed properties.

Thanks Hynek. Just a short question, could you tell me please what is a referencing indexed properties?

#54 - 01/08/2020 09:00 AM - Hynek Cihlar

Marius Gligor wrote:

> could you tell me please what is a referencing indexed properties?

I meant an assignment of an indexed property to a variable and passing an indexed property to a procedure or function invocation.

#55 - 01/09/2020 03:29 AM - Marius Gligor

I tested and fixed referenced indexed properties conversion rules. Basically the rules determine when to use set or get method. I used the following UAST test:

```
DEFINE VARIABLE lColors AS LOGICAL NO-UNDO.
DEFINE VARIABLE CtrlFrame AS WIDGET-HANDLE NO-UNDO.
DEFINE VARIABLE chCtrlFrame AS COM-HANDLE NO-UNDO.
DEFINE VARIABLE iNode1 AS INTEGER NO-UNDO.
DEFINE VARIABLE iNode2 AS INTEGER NO-UNDO.
DEFINE VARIABLE bgColor AS INTEGER NO-UNDO.

PROCEDURE GetBgColors :
    DEFINE INPUT PARAMETER pColor AS INTEGER.
    MESSAGE pColor.
END PROCEDURE.

IF lColors THEN
    ASSIGN chCtrlFrame:tree1:NodeBgColor(1) = COLOR-TABLE:GET-RGB-VALUE(15)
           chCtrlFrame:tree1:NodeFgColor(2) = COLOR-TABLE:GET-RGB-VALUE(1).

IF lColors THEN
    ASSIGN chCtrlFrame:tree1:NodeBgColor(iNode1) = COLOR-TABLE:GET-RGB-VALUE(15)
           chCtrlFrame:tree1:NodeFgColor(iNode2) = COLOR-TABLE:GET-RGB-VALUE(1).

bgColor = chCtrlFrame:tree1:NodeBgColor(iNode2).
RUN GetBgColors chCtrlFrame:tree1:NodeBgColor(iNode2).
```

which is converted in Java as:

```
 if ((lColors).booleanValue())
```

```
 {
    hTree.unwrapTreeFace().setNodeBgColor(new integer(1), ColorTable.getRgbValue(15));
    NoopCom.handle().chain("tree1").setIndexedProperty("NodeFgColor", ColorTable.getRgbValue(1), ComParameter.
input(2));
 }

 if ((lColors).booleanValue())
 {
    hTree.unwrapTreeFace().setNodeBgColor(iNode1, ColorTable.getRgbValue(15));
    NoopCom.handle().chain("tree1").setIndexedProperty("NodeFgColor", ColorTable.getRgbValue(1), ComParameter.
input(iNode2));
 }

 bgColor.assign(hTree.unwrapTreeFace().getNodeBgColor(iNode2));
 ControlFlowOps.invoke("GetBgColors", hTree.unwrapTreeFace().getNodeBgColor(iNode2));
```

Changes has been committed in branch 3856b rev 11363
Hynek, please do a review of my changes.

#56 - 01/09/2020 03:34 AM - Hynek Cihlar

Code review 3856b revision 11363. The changes look good.

#57 - 01/10/2020 07:20 AM - Sergey Ivanovskiy

- Related to Feature #4174: implement calendar control/dtpicker OCX replacement added

#58 - 01/14/2020 02:31 PM - Sergey Ivanovskiy

Marius, could you explain why ENABLED and VALUE progress bar properties were named as PB-ENABLED and PB-VALUE? Did you encounter any conversion conflicts if these properties were without PB- prefix?

#59 - 01/14/2020 02:47 PM - Greg Shah

There are existing 4GL attributes of the same names, so the PB- prefix was needed to differentiate these from the "built-in" 4GL versions.

#60 - 01/14/2020 03:59 PM - Sergey Ivanovskiy

- File 3856b_datetime_picker.patch added

Sergey Ivanovskiy wrote:

> Marius, could you explain why ENABLED and VALUE progress bar properties were named as PB-ENABLED and PB-VALUE? Did you encounter any conversion conflicts if these properties were without PB- prefix?

OK. I was writing at this moment when you were answering my question.

If I understand correctly, then the problems appear when common OCX control properties must be resolved as widget's handle properties as in the case of VALUE or ENABLED. To resolve some conversion rules interferences we should define new names and add new methods aliases.
Thus after these changes I was able to map such strings chCtrlFrame:DTPicker:VALUE into hDTPicker.unwrapDateTimePicker().getDTPValue() in the presence of this hint file

```
<extension>
   <ocx control-frame="CtrlFrame" control-frame-handle="chCtrlFrame" com-handle="chCtrlFrame:DTPicker" target-handle="hDTPicker" target-widget="datetime-picker" />
   <drop-procedure name="control_load" />
</extension>
```

Are there another ways to deal with these issues?

#61 - 01/14/2020 04:17 PM - Greg Shah

The problem here is that handle has no strong typing information that can always be known at parsing time. Sometimes code operates on a well known system-handle or on a handle that was assigned by some preceding 4GL code. But these typing approaches are limited. Since a single handle instance can be used for any number of different types, the 4GL code can be written such that typing is not possible. When handle instances are passed through multiple levels as parameters, it often occurs that any remaining type clues are lost.

The way we deal with this problem is that we create shared interfaces for any attribute or method that is implemented in differently typed objects. We then can unwrap the handle to the common interface and be sure that it will work. This means that if you want to continue using the original keyword, then you will have to make a shared interface that is implemented in all widgets/resources that provide the attribute or method. This will only work if the types and API are compatible. If there are different types or signatures used, then it cannot be done without more type/signature analysis at conversion time.

Making the names separated is a sure way to solve the issue.

#62 - 01/16/2020 08:47 AM - Marius Gligor

I've fixed regression test errors as follow:
- remove UIB_S = chCtrlFrame:LoadControls(OCXFile, "CtrlFrame":U) instead noop-com
BTW as I understood drop-procedure is no longer used. Should we remove conversion rules for dropping or keep them as an option?
- fixed composed properties like font:name
- fixed conversion of component-handle into handle for statements like:

```
PROCEDURE test:
   DEFINE INPUT PARAMETER chCtrl AS COMPONENT-HANDLE.

   chCtrl:GetCellStringValue(1, 2, output "text").
   chCtrl:GetFirstChildNode(1, 1, "text").
   chCtrl:GetNextSiblingNode(1, 1, "text").
END PROCEDURE.
```

and ext-hints like <ocx com-handle="chCtrl" target-handle="hTree" target-widget="treeview" />

I converted many uast tests and I found no problems. However, Hynek when you have time please take a look over my changes.
Changes has been committed into branch 3856b rev 11365-11371
Next I'm going to prepare a new test environment for a large project and run the conversion once again.

#63 - 01/20/2020 02:41 AM - Hynek Cihlar

Code review 3856b revisions 11365-11371. The changes look good.

#64 - 01/20/2020 07:25 AM - Greg Shah

Marius: Please rebase 3856b from the latest trunk. My plan is for 3856b to be the next branch to merge to trunk. This means we need to do a final round of testing on the rebased branch, both conversion and runtime.

#65 - 01/22/2020 09:49 AM - Marius Gligor

I created a new task branch 3856c from current trunk, then I merged changes from 3856b onto 3856c rev. 11341

#66 - 01/23/2020 03:49 PM - Sergey Ivanovskiy

Marius Gligor wrote:

> I created a new task branch 3856c from current trunk, then I merged changes from 3856b onto 3856c rev. 11341

Are you planning to merge 3856b into the trunc? 4174a is over the current 3856b rev 11371.

#67 - 01/23/2020 04:36 PM - Greg Shah

No, 3856b has been merged into 3856c instead of rebasing. 3856c is now the branch for this task. Marius is regression testing this branch now. If 3856c passes, it will be merged to trunk.

#69 - 01/29/2020 05:30 PM - Greg Shah

- % Done changed from 0 to 100

- Status changed from Review to Closed

Task branch 3856c was merged to trunk as revision 11341. 3856c was then archived.

#70 - 01/29/2020 05:34 PM - Greg Shah

Branch 3856b was dead archived.

Files

abl-com-object-viewer.png	14.3 KB	11/01/2019	Marius Gligor
3856b_datetime_picker.patch	9.34 KB	01/14/2020	Sergey Ivanovskiy