# Base Language - Feature #4395

## add support for KEEP-MESSAGES option

11/09/2019 08:01 AM - Greg Shah

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

**Description**

**Related issues:**

| | |
|---|---|
| Related to Base Language - Bug #4628: redirection doesn't seem to close an ou... | **New** |

### History

#### #1 - 11/09/2019 08:15 AM - Greg Shah

For sure, this can be used in OUTPUT TO and must be implemented for that case.

This option is only documented as possible for OUTPUT TO but it has been seen in INPUT FROM as well. Currently, the parser drops this option when matched in INPUT FROM. The application being converted in this case DOES use this. We need to test if this has real meaning and if so, allow the option to be emitted into the tree (and then implement the proper support for conversion/runtime).

#### #2 - 04/15/2020 09:58 AM - Greg Shah

*- Assignee set to Roger Borrello*

#### #3 - 04/17/2020 12:33 PM - Roger Borrello

```
OUTPUT [ STREAM stream | STREAM-HANDLE handle ] TO
 { PRINTER [ printer-name ]
   | opsys-file
   | opsys-device
   | TERMINAL
   | VALUE ( expression )
   | "CLIPBOARD"
 }
```

[ **KEEP-MESSAGES** ]

Causes the following messages not to echo to the default window: ABL error and warning messages, and messages from the MESSAGE statement. If you specify KEEP-MESSAGES, these messages are sent only to the output stream you specify.

MESSAGE statements using the VIEW AS ALERT BOX option are an exception. The KEEP-MESSAGES option does not suppress the message box if there is a way available to display it.

#### #4 - 04/17/2020 02:27 PM - Roger Borrello

```
INPUT [ STREAM stream | STREAM-HANDLE handle ] FROM
  {   opsys-file
    | opsys-device
    | TERMINAL
    | VALUE ( expression ) [ NO-ATTR-LIST ]
  }
```

No mention of KEEP-MESSAGES.

GUI code example:

```
output to value(cLogDir) keep-messages append unbuffered no-echo no-map no-convert.
put unformatted cDebugMessage skip.
output close.
```

```
  output to value(cErrorTestFile).
  put "OK":u skip.
  input from os-dir(cDir) keep-messages no-echo.
  input close.
  output close.
```

#### #5 - 04/17/2020 03:49 PM - Roger Borrello

Updated testcase uast/binary/read-text-and-binary.p to show there is no differences in having keep-messages on the input from line and not having it on the line. It can be ignored.

I have also updated testcase uast/io/basic_redirected_output.p to validate the behavior of output to ... keep-messages.

#### #6 - 04/17/2020 05:18 PM - Roger Borrello

The input_output.rules do not do anything with the KEEP-MESSAGES options. There is the possibility of adding an additional parameter to StreamFactory.openFileStream constructor. The constructor would set the keepMessages flag for other I/O methods to use, to determine if the

message should be echoed to the default window. These are 4GL errors and warnings, and MESSAGE statements. But I am not sure whether to StreamFactory is the best class to hold the keepMessages flag.

Wouldn't this be better stored with LogicalTerminal?

#### #7 - 04/17/2020 06:39 PM - Greg Shah

Please look at StreamFactory. You will see that it is stateless in regard to new streams. The KEEP-MESSAGES flag would not be there, but would be passed to the constructor for the stream itself. The flag needs to be in the stream and it needs to be one of the flags that gets actually pushed down to the remote side. This means there needs to be a setter in RemoteStream. See RemoteStream.setBinary() for an example.

The server-side worker method for MESSAGE is LT.message(Object[], boolean, Accessor, boolean, String, ColorSpec, handle). But I think the real honoring of KEEP-MESSAGES will be on the client. The ThinClient.message() that is called is just there to call message() on the right window instance. You can see it by searching in TC for MessageReturnValue. In Window, the initial method called is public MessageReturnValue message(String, boolean, BaseDataType, boolean, String, Color) but the output part is done in public int message(String, Color, boolean). I suspect that is where the change is needed. You'll have to get the Stream instance and check the isKeepMessages() to know if you need to bypass the message area output.

#### #8 - 04/17/2020 06:49 PM - Constantin Asofiei

Greg/Roger: the IO options can be emitted as a setter; see this portion of convert/input_output.rules:

```
      <!-- set the options into this stream -->
      <rule>parent.type == prog.io_options
```

```
         <rule>type == prog.kw_portrait
            <action>
                refid = createJavaAst(java.method_call, "setPortrait", parentid)
            </action>
         </rule>
```

I would just call the setter directly as setKeepMessages(true), instead of modifying the StreamFactory APIs.

#### #9 - 04/18/2020 06:40 AM - Greg Shah

> I would just call the setter directly as setKeepMessages(true), instead of modifying the StreamFactory APIs.

Agreed, this is better.

#### #10 - 04/19/2020 09:58 AM - Roger Borrello

That part is straightforward. I am using setKeepMessages() without a parameter because it is an option that defaults to off, and KEEP-MESSAGE can only turn it on. There isn't a language construct to turn it off like there is for ECHO and NO-ECHO.

With respect to where setKeepMessages/getKeepMessages should be implemented, I added it to Stream, similar to how setEcho/getEcho is implemented. The other part is where getKeepMessages needs to be utilized to determine when to suppress 4GL error and warning messages, as well as the normal message statements.

#### #11 - 04/19/2020 10:16 AM - Roger Borrello

It looks like there is a bug in LogicalTerminal.message() whereby the output stream is never checked for redirection, unless the input stream is redirected.

```
      Stream in  = UnnamedStreams.input();
      Stream out = UnnamedStreams.output();

      if (in != null)
      {
         // possible stream reading mode
         if (var != null)
         {
            in.resetCurrentLine();

            try
            {
               // read from the stream
               if (!GenericFrame.getField(in, var, null, null, format, true))
               {
                  // failure in silent error mode
                  return;
               }
            }

            finally
            {
               in.resetCurrentLine();
            }

            if (out == null)
            {
               // output is not redirected, we are done
               return;
            }
         }
      }
```

Shouldn't the if (out == null) be outside the check for if (in != null) or can it only follow that output is redirected if input is (which seems odd to me)?

#### #12 - 04/19/2020 10:35 AM - Greg Shah

getKeepMessages() should be named isKeepMessages() since the flag is a boolean.  The standard for Java "bean" names (properties with getters/setters) uses get for non-boolean values and is for boolean values.

> Shouldn't the if (out == null) be outside the check for if (in != null) or can it only follow that output is redirected if input is (which seems odd to me)?

It is fine.  Notice that we aren't doing anything else with out in that method.  All output redirection processing is happening on the client side.  As noted above, the Window class is where your changes will be needed.

#### #13 - 04/19/2020 05:14 PM - Roger Borrello

Greg Shah wrote:

> getKeepMessages() should be named isKeepMessages() since the flag is a boolean.  The standard for Java "bean" names (properties with getters/setters) uses get for non-boolean values and is for boolean values.

I do have Stream modified to have setKeepMessages() and isKeepMessages(). How does the flag get to the client? Do I still need to fiddle with RemoteStream if I am converting as:

```
            if ((keepmsg).booleanValue())
            {
               UnnamedStreams.assignOut(StreamFactory.openFileStream("./test.txt", true, false));
               UnnamedStreams.safeOutput().setKeepMessages();
            }
            else
            {
               UnnamedStreams.assignOut(StreamFactory.openFileStream("./test.txt", true, false));
            }
```

#### #14 - 04/19/2020 05:45 PM - Greg Shah

> Do I still need to fiddle with RemoteStream if I am converting as:

Yes.  RemoteStream is how it gets to the client.

#### #15 - 04/20/2020 01:02 AM - Roger Borrello

Greg Shah wrote:

> > Do I still need to fiddle with RemoteStream if I am converting as:
>
> Yes.  RemoteStream is how it gets to the client.

I have that implemented in the Stream, RemoteStream, LowLevelStream, and StreamDaemon. But I'm not sure what to do with Window and ThinClient in order to have access to the flag.

#### #16 - 04/20/2020 07:22 AM - Greg Shah

See ThinClient.getCurrentRedirection().

#### #17 - 04/20/2020 09:27 AM - Roger Borrello

Greg Shah wrote:

> See ThinClient.getCurrentRedirection().

On the server side, RemoteStream.setKeepMessages is called from business logic:

```
public void setKeepMessages()
{
   super.setKeepMessages();
   work.obtain().rss.setKeepMessages(id);
}
```

Stream.setKeepMessages, sets the instance variable to true (keepMessages in Stream). I cannot tell what happens with work.obtain().rss.setKeepMessages(id) as it's goes through a proxy call. I do know my breakpoint in StreamDaemon.setKeepMessages(int id) doesn't get hit.

At that point, I'm not sure how and where the "handshake" occurs. I have added setKeepMessages and isKeepMessages methods in ThinClient, since the getCurrentRedirection() sample uses isRedirected to observe the state of the currentStream. It's not quite the same type of passing of an option to the client.

#### #18 - 04/20/2020 09:30 AM - Greg Shah

> I do know my breakpoint in StreamDaemon.setKeepMessages(int id) doesn't get hit.

This breakpoint will be hit in the client, not in the server.

> I have added setKeepMessages and isKeepMessages methods in ThinClient, since the getCurrentRedirection() sample uses isRedirected to observe the state of the currentStream. It's not quite the same type of passing of an option to the client.

I don't think you need this. Why would you ever call setKeepMessages() in ThinClient? It is already set/not-set from the server. You can call isKeepMessages() directly on any non-null returned Stream instance from getCurrentRedirection() and voila, you are good to go.

#### #19 - 04/20/2020 11:16 AM - Roger Borrello

Now we are cooking... keepMessages is now available to the client in the critical location.

At looking at the specification, we are to hold off on echoing to the default window when dealing with ABL error and warning messages, and messages from the MESSAGE statement. Also, when VIEW AS ALERT BOX option is set, KEEP-MESSAGES does **not** suppress the message box if there is a way available to display it.

My current understanding is that MESSAGE is handled by Window.message and alert boxes are handled by ThinClient.messageBox. I am looking at the proper place to determine whether or not to suppress the output to the default windows, but some direction on whether or not ABL errors and warnings are handled via a different interface would help me complete the task.

#### #20 - 04/20/2020 11:40 AM - Greg Shah

In the spirit of "learning to fish", you could use the following to answer the question yourself:

my-int-var = integer("garbage").

Set your breakpoint in both places (ON THE CLIENT) and see which is used.

Hint: Try this in both ChUI and GUI please. You may find the results to be different.

#### #21 - 04/20/2020 01:45 PM - Roger Borrello

Ready for review.

```
Committing to: /home/rfb/secure/code/p2j_repo/p2j/active/4231b/

modified rules/convert/input_output.rules
modified src/com/goldencode/p2j/ui/chui/ThinClient.java
modified src/com/goldencode/p2j/ui/client/Window.java
modified src/com/goldencode/p2j/ui/client/gui/WindowGuiImpl.java
modified src/com/goldencode/p2j/util/LowLevelStream.java
modified src/com/goldencode/p2j/util/RemoteStream.java
modified src/com/goldencode/p2j/util/Stream.java
modified src/com/goldencode/p2j/util/StreamDaemon.java
Committed revision 11470.
```

Updated uast/io/basic_redirected_output.p testcase to handle the possibilities.

An unrelated issues has to do with how display "Last Line" with frame f-last. is handled, with message "A message.". following it. There seems to be a line feed missing. This happens if the OPSYS value in **p2j.cfg.xml** is UNIX or WIN32.

```
The TITLE line!

         4 the first text                 04/20/20
        19 more stuff here               10/10/49
       300 that's all folks              11/01/99

Last LineA message.
** Invalid character in numeric input g. (76)
```

In 4GL, there is a line feed.

```
E:\testcases\uast>type test.txt

The TITLE line!

         4 the first text                 04/20/20
        19 more stuff here               10/10/49
       300 that's all folks              11/01/99

Last Line
A message.
** Invalid character in numeric input g. (76)
```

Also, in GUI, we are pre-filling the entry field shown on message "Redirected?" set redir as logical. with "no", rather than blank, in 4GL.

**#22 - 04/20/2020 01:45 PM - Roger Borrello**

*- Status changed from New to WIP*

*- % Done changed from 0 to 100*

**#23 - 04/20/2020 02:01 PM - Greg Shah**

> An unrelated issues has to do with how display "Last Line" with frame f-last. is handled, with message "A message.". following it. There seems to be a line feed missing.
>
> Also, in GUI, we are pre-filling the entry field shown on message "Redirected?" set redir as logical. with "no", rather than blank, in 4GL.

These are 2 separate issues. Assuming they both exist prior to your changes, you can create a new task for each one, in the UI project.

**#24 - 04/20/2020 02:23 PM - Greg Shah**

Code Review Task branch 4231b Revision 11470

1. In KEEP-MESSAGES mode, have you checked the following behavior?

- Does a MESSAGE statement cause a hidden window to be made visible? This would only affect GUI since ChUI only has one window and you cannot hide it.
- Is the message area cleared even though no new messages are displayed? This may only affect ChUI. This will require at least one item in the message area to check.
- Is there a pause before hiding any old output? This may only affect ChUI. This will require at least two lines written in the message area to check. For example, in normal conditions in ChUI, if you have 3 message statements in a row then there is a pause ("Press any key to continue.") in between the 2nd nd 3rd messages.

Your implementation leaves all of these intact, but I wonder if any of them is actually still needed.

**#25 - 04/20/2020 02:27 PM - Greg Shah**

Code Review (cont.)

2. In LowLevelStream and StreamDaemon you need to add an id parameter javadoc entry for setKeepMessages(int id).

**#26 - 04/20/2020 02:32 PM - Roger Borrello**

The latter seems most likely the same issue as #4619.

**#27 - 04/20/2020 03:31 PM - Roger Borrello**

*- File Screenshot from 2020-04-20 15-28-45.png added*

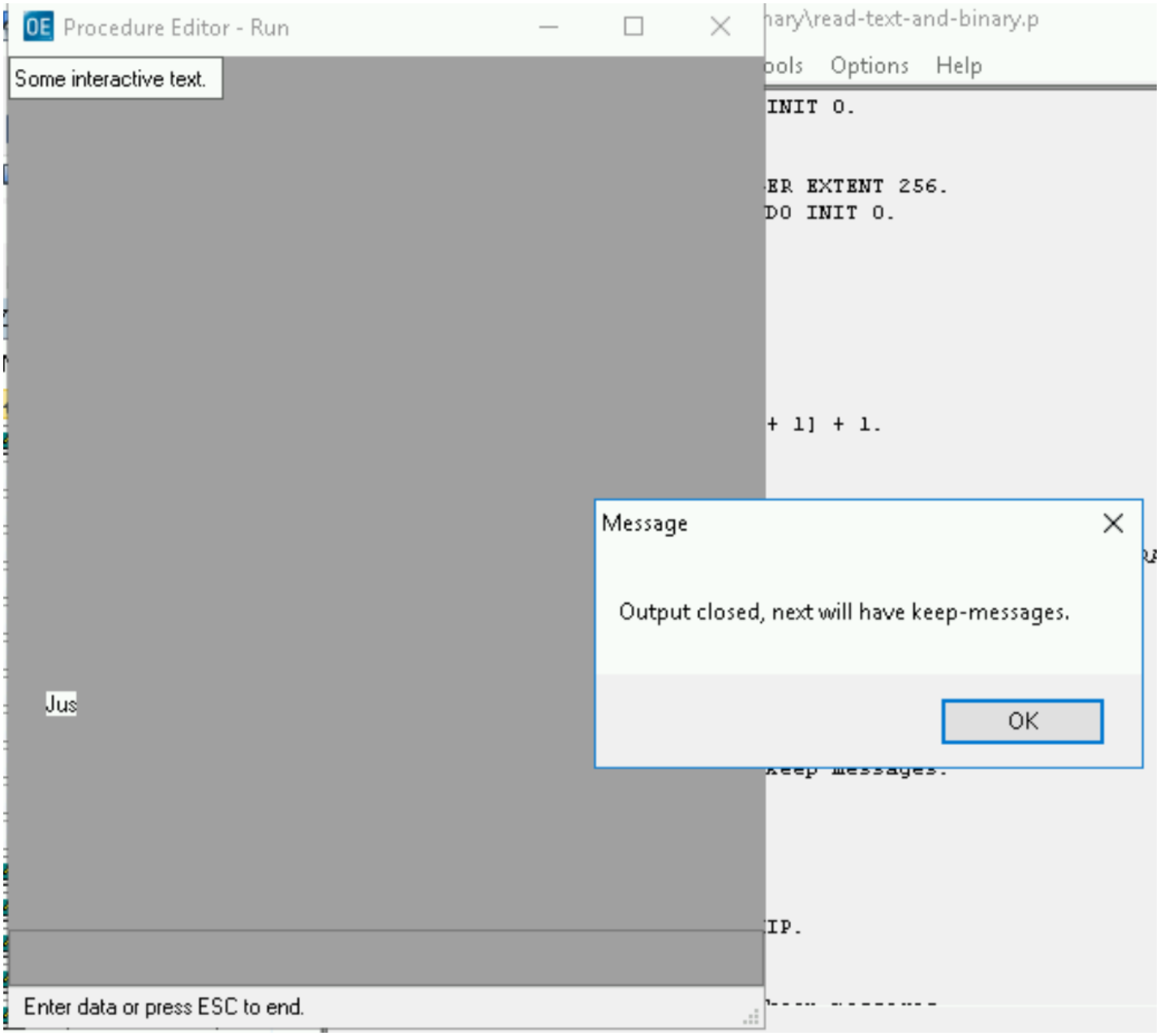
*- File Screenshot from 2020-04-20 15-27-59.png added*

I am not seeing the testcases put screen... show up in GUI like it does in 4GL. In ThinClient, the putScreen method has:

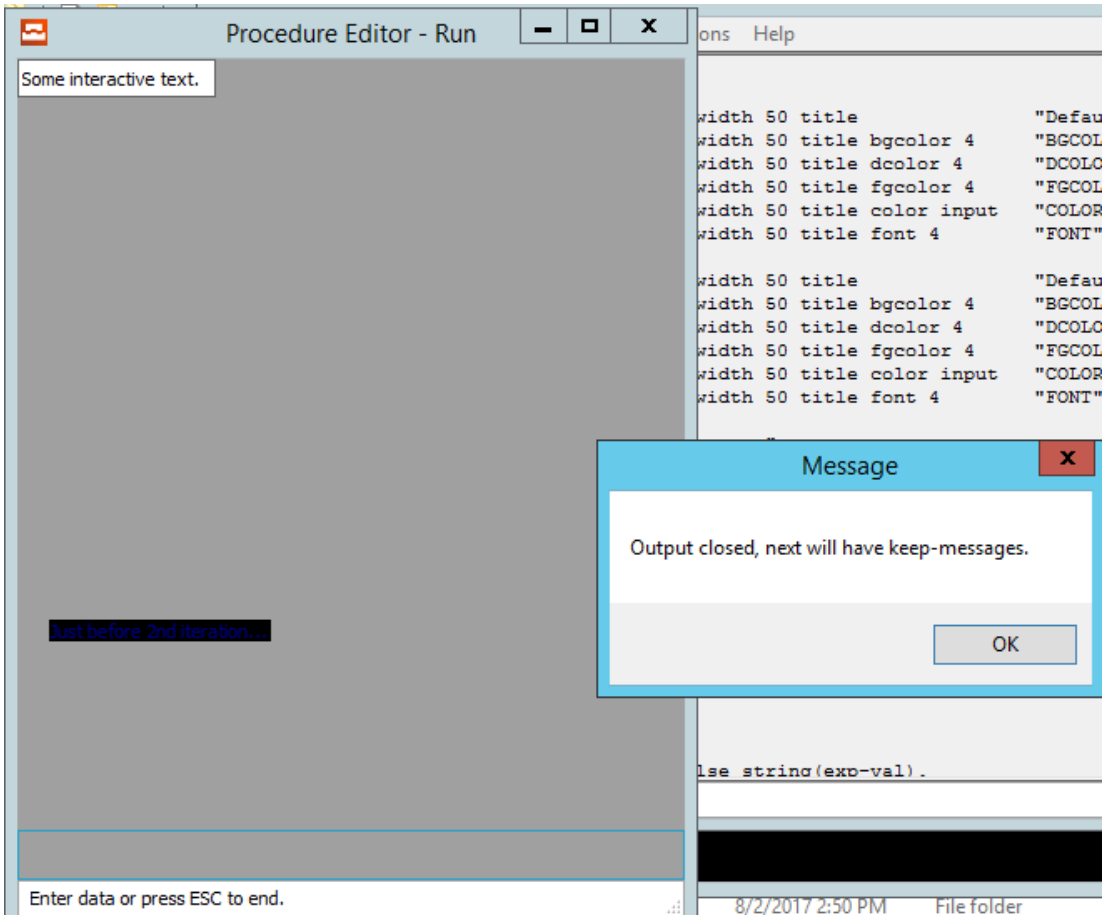
```
   public void putScreen(String text, double row, double column, Color color)
   {
      if (!isChui())
      {
         // in GUI, PUT SCREEN is a no-op
         return;
      }
```

I can see where the documentation indicates ChUI only. But it does show up in GUI.

GUI 10.2B



GUI 11.6.3




**#28 - 04/20/2020 03:35 PM - Roger Borrello**

*- File Screenshot_2020-04-20_15-28-45.png added*

*- File Screenshot_2020-04-20_15-27-59.png added*

#### #29 - 04/20/2020 03:38 PM - Constantin Asofiei

Roger, PUT SCREEN issues in GUI are known, see #2372.

#### #30 - 04/20/2020 04:53 PM - Roger Borrello

I have updates to correct behavior where Press space bar to continue was displayed in ChUI, even though output was redirected.

One of the behaviors of the 4GL and FWD when the error condition raised from i = integer("garbage"). is to jump execution from the beginning, and the statements after that are not executed. When I added no-error, it created a situation where no errors were logged in the redirection. All good so far, but in FWD, one of the lines that isn't processed after the ** Invalid character in numeric input g. (76) is thrown is the output close. Because of this, the stream is still maintaining the KEEP-MESSAGES option, and when the program re-executes the message "Redirected?" set redir as logical. the text is suppressed from being displayed.

It's difficult to determine how the 4GL is actually handling this. Perhaps it implicitly closes the stream, and we don't?

#### #31 - 04/21/2020 08:48 AM - Greg Shah

> It's difficult to determine how the 4GL is actually handling this. Perhaps it implicitly closes the stream, and we don't?

Well, we used to do this for sure.  I know of no reason it would not work, but it is possible to have broken.  On the other hand, I would expect our ChUI regression testing to break badly if this was not working.  So it would be a surprise.

> Because of this, the stream is still maintaining the KEEP-MESSAGES option, and when the program re-executes the message "Redirected?" set redir as logical. the text is suppressed from being displayed.

What is the testcase?  Why is it retrying instead of exiting?

#### #32 - 04/21/2020 09:16 AM - Roger Borrello

Greg Shah wrote:

> What is the testcase?  Why is it retrying instead of exiting?

It is uast/io/basic_redirected_output.p and behaves similarly in 4GL. That is, it seems to restart from the beginning upon executing the invalid i = integer("bogus"). only the output stream is handled better (it closes first).

#### #33 - 04/21/2020 10:07 AM - Greg Shah

Don't change existing sample code in a way that destroys the original intent and control flow.  Add a new testcase and restore that program.

Also, a simpler testcase will make it more clear where the problem lies.  Perhaps there is an implicit stream close on retry which we are not supporting.  Stream implements Finalizable.  You can see how we implicitly close streams upon exiting the scope in which they were defined.  See Stream.finished(), Stream.retry(), Stream.iterate() etc...

I'm confident the Stream.iterate() should be a no-op, but perhaps retry() needs to close (and maybe flush too, you'll have to check it).

#### #34 - 04/21/2020 03:29 PM - Roger Borrello

Greg Shah wrote:

> Code Review Task branch 4231b Revision 11470
>
> 1. In KEEP-MESSAGES mode, have you checked the following behavior?
>
>    - Does a MESSAGE statement cause a hidden window to be made visible?  This would only affect GUI since ChUI only has one window and you cannot hide it.

Added testcase uast/window/redirected_message_to_hidden_window.p which shows that we are behaving in the same manner as 4GL. That is, even if you redirect output and KEEP-MESSAGES, MESSAGE will pop a window that is hidden.

#### #35 - 04/24/2020 03:57 PM - Roger Borrello

*- Related to Bug #4628: redirection doesn't seem to close an output stream on a block retry added*

#### #36 - 04/29/2020 03:19 PM - Roger Borrello

Task is ready for test.

#### #37 - 04/30/2020 10:09 AM - Greg Shah

*- Status changed from WIP to Test*

#### #38 - 06/20/2020 12:00 PM - Greg Shah

*- Status changed from Test to Closed*

Task branch 4231b has been merged to trunk as revision 11347.

### Files

| | | | |
|---|---|---|---|
| Screenshot from 2020-04-20 15-28-45.png | 18 KB | 04/20/2020 | Roger Borrello |
| Screenshot from 2020-04-20 15-27-59.png | 58.4 KB | 04/20/2020 | Roger Borrello |
| Screenshot_2020-04-20_15-28-45.png | 18 KB | 04/20/2020 | Roger Borrello |
| Screenshot_2020-04-20_15-27-59.png | 58.4 KB | 04/20/2020 | Roger Borrello |