Base Language - Bug #2507

format validation with editing blocks

01/28/2015 04:04 AM - Constantin Asofiei

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	GUI Support for a Complex ADM2 App		
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 01/28/2015 04:05 AM - Constantin Asofiei

Constantin,

Consider this:

```
def var txt1 as char.
def var txt2 as char.

update txt1 format "999"
       txt2 format "x(3)"
       editing:
          readkey.
          apply lastkey.
       end.
```

The "999" character format requires that there be 3 numeric digits in the resulting edited value. If not, then even in the 4GL we get an error.

The problem here is that when we type the 1st digit char, the editing block gets invoked, the char is read via readkey and then we apply it to the client. The following is the call sequence that causes the problem:

```
ThinClient.apply ()
ThinClient.getScreenBuffer ()
FillIn.getValue ()
FillIn.update ()
StringFormat$CharBuf.checkFormat ()
```

At the very end of the apply, we update the screen buffer to send the edited results back to the server. As part of that, we call update() which checks the validity of the data based on the format string. The problem is that we are in mid-edit here and the overall result can't possibly ever pass this check. So the code generates an error 631 and an error 629 and forces the cursor to the first char position in the field.

The 4GL has no such behavior, it only validates against the format at the very end of the edit (when you try to leave the field via TAB, ENTER or F1).

I suspect this has been broken for a while (forever?).

Thoughts?

Thanks,
Greg

#2 - 01/28/2015 04:19 AM - Constantin Asofiei

- File ca_upd20150128a.zip added

From what I see, the format validation is done when LEAVE is sent for the widget. More, when EDITING mode is enabled, the buffer is sent to the server-side, but no validation is done.

Attached update fixes this. Runtime testing is pending.

#3 - 01/28/2015 08:31 AM - Greg Shah

Code Review ca_upd20150128a.zip

My only concern is with the use of checkFormat() instead of validateFormat() on LEAVE. The 4GL does indeed generate a single error 630 instead of the combination of 631 and 629 for character. In addition both DateFormat and DatetimeFormat have some very specific processing that is implemented in validateFormat() which will no longer be executed on LEAVE.

Other than that, the changes are good.

#4 - 01/28/2015 11:55 AM - Constantin Asofiei

Greg Shah wrote:

> Code Review ca_upd20150128a.zip
>
> My only concern is with the use of checkFormat() instead of validateFormat() on LEAVE. The 4GL does indeed generate a single error 630 instead of the combination of 631 and 629 for character.

I think we are going into murky waters again. Looks like validation is done only if the buffer changes. Use this:

```
def var txt1 as char.
def var txt2 as char.

display "aaa" @ txt1.

set txt1 format "999" txt2 format "x(3)".
```

and press ENTER twice: this will allow a LEAVE at txt1 (first ENTER) and the second ENTER (frame LEAVE?) will force the txt1's screen-value to be validated against the format, showing a 630 "character a at position 1 must be digit." and a 143 "unable to evaluate field for assignment" message.

If you change the text to "aa1" and press ENTER, it will show you only one 630 error. This is different from the case in note 1, when UPDATE is used with an incomplete value, like "12" instead of "123" - it will show both 630 and 629 error messages.

I think 629 error appears only if all chars are valid or blank, but the field is not fully input (some chars are blank).

> In addition both DateFormat and DatetimeFormat have some very specific processing that is implemented in validateFormat() which will no longer be executed on LEAVE.

Yes, I missed this.

With this in mind, I'm still trying to determine why both checkFormat and validateFormat APIs in DisplayFormat$Presentation were needed...

#5 - 01/29/2015 10:52 AM - Constantin Asofiei

- File ca_upd20150129b.zip added

This is replacement of 0128a.zip. Enhances the support for character formats and is a good step forward, but there are still some missing cases:

1. if the fill-in is empty and the key is not accepted by the fill-in's format, then the fill-in must not be marked as "needsFormatCheck"
2. date validation when something like this is used:

   ```
   def var date-var as date format "99/99/9999".
   display "bogustexts" @ date-var.
   set date-var.
   ```

   will not work OK - the fill-in gets reset in P2J on the first key, which is not how 4GL works.

This is going through runtime testing now. Will release after it passes testing, as is a good step forward.

#6 - 01/29/2015 03:57 PM - Constantin Asofiei

> This is going through runtime testing now.

CTRL-C passed, MAIN part was killed and restarted, as after ~5 hours was still running.

#7 - 01/29/2015 04:28 PM - Greg Shah

Code Review ca_upd20150129b.zip

I'm fine with the changes.

#8 - 01/30/2015 01:11 AM - Constantin Asofiei

Greg Shah wrote:

> Code Review ca_upd20150129b.zip
>
> I'm fine with the changes.

Passed runtime testing, released to rev 10728.

#9 - 01/30/2015 05:21 AM - Greg Shah

- Target version set to Milestone 12

- Status changed from WIP to Closed

#10 - 11/16/2016 12:13 PM - Greg Shah

- Target version changed from Milestone 12 to GUI Support for a Complex ADM2 App

Files

ca_upd20150128a.zip	51.6 KB	01/28/2015	Constantin Asofiei
ca_upd20150129b.zip	66 KB	01/29/2015	Constantin Asofiei