Database - Bug #2570

buffers must be members of the class which defines their tightest scope

05/05/2015 11:46 AM - Eric Faulhaber

Status:	Closed	**Start date:**	05/05/2015
Priority:	Normal	**Due date:**	
Assignee:	Eric Faulhaber	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Cleanup and Stablization for Server Features		
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 05/05/2015 12:10 PM - Eric Faulhaber

- File ecf_upd20150505a.zip added

P2J incorrectly creates all record buffers as instance members of the top level class. Buffers scoped more tightly than this must instead convert to instance members of the inner class which matches their scope. For example, see the following program (testcases/uast/recursive-intproc-local-buffer.p):

```
define var v1 as integer.
define temp-table tt1
   field f1 as int.
define var v2 as integer.

procedure int-proc-temp:

   define var i1 as integer.
   define buffer tt1 for tt1.
   define var i2 as integer.

   v1 = 6.
   create tt1.
   tt1.f1 = i1.
   create tt1.
   tt1.f1 = i2.

   run int-proc-temp.

end.

procedure int-proc-perm:

   define var v1 as integer.
   define buffer book for book.
   define var i2 as integer.

   v1 = 7.
   create book.
   book.book-id = v1.
   create book.
   book.book-id = i2.

   run int-proc-perm.

end.

procedure int-proc-param:
```

```
   define input parameter table for tt1.

   find first tt1.

end.

find first book where book.book-id = v2 no-lock.
```

Currently, all buffers defined in these internal procedures generate top-level class instance variables. These instance variables are set to null in their declarations, and are initialized in an anonymous constructor generated within the anonymous inner subclass of Block which is passed to BlockManager.internalProcedure. This causes a problem when an internal procedure calls itself recursively, since each call resets the reference to the instance variable for the record buffer.

The solution is to create the record buffer instance variables at the anonymous inner subclass of Block. As a nice side effect, we can eliminate most (all?) of the buffer renaming which causes a proliferation of names with "Buf2", "Buf3", ... "BufN" suffixes, since these generally are the result of artificial scope conflicts which stem from making all the buffer names global.

Greg: please review the attached, proposed update. I'm running conversion regression testing on it now.

#2 - 05/05/2015 04:17 PM - Eric Faulhaber

There were regressions with naming conflicts. Working on a fix.

#3 - 05/06/2015 03:58 AM - Eric Faulhaber

- File ecf_upd20150506a.zip added

I've decided to defer the name conflict improvements indefinitely, as it turns out there's a lot of potential breakage that this can cause. Also, the improvement would only be incremental; the disambiguation still needs to take place in more cases than I realized. The more important issue is getting the record buffers defined at the right scope. The attached update accomplishes this, with the name conflict changes (mostly) rolled back. It supersedes my 0505a update. It has passed conversion regression testing and is undergoing runtime test.

Greg: please review.

#4 - 05/06/2015 10:04 AM - Greg Shah

Code Review ecf_upd20150506a.zip

I don't see much in the way of problems. Where there many changes to the MAJIC conversion results?

1. In buffer_definitions.rules, it would be useful to expand the comment <!-- non-parameter explicit shared approach --> to explain that shared buffers cannot be defined in internal procedures, triggers (?) or functions. This will help readers understand why the scoped buffer definition processing doesn't apply there.

2. I assume that the bufname.startsWith("*can-find-") being replaced with isNote("canfind") is a safe thing to do (this is in buffer_name_conflicts.rules

and cross_namespace_conflicts.rules). Was this a latent bug after the recent CAN-FIND changes?

3. In the same places as the CAN-FIND changes, you also removed the "refid" annotation check. I assume that was intentional and OK?

#5 - 05/06/2015 12:31 PM - Eric Faulhaber

Greg Shah wrote:

> Code Review ecf_upd20150506a.zip
>
> I don't see much in the way of problems. Where there many changes to the MAJIC conversion results?

No, only those expected (moving certain buffer declarations/definitions from the external class' instance variables to an internal procedure's anonymous inner Block subclass).

> 1. In buffer_definitions.rules, it would be useful to expand the comment <!-- non-parameter explicit shared approach --> to explain that shared buffers cannot be defined in internal procedures, triggers (?) or functions. This will help readers understand why the scoped buffer definition processing doesn't apply there.

Thanks for the explanation. I'll add it.

> 2. I assume that the bufname.startsWith("*can-find-") being replaced with isNote("canfind") is a safe thing to do (this is in buffer_name_conflicts.rules and cross_namespace_conflicts.rules). Was this a latent bug after the recent CAN-FIND changes?

Yes. I didn't notice it manifest anywhere, but better to clean it up.

> 3. In the same places as the CAN-FIND changes, you also removed the "refid" annotation check. I assume that was intentional and OK?

It was intentional (I read it as being part of the CAN-FIND check). However, I did have issues converting the customer app, so I have to revisit the update...

#6 - 05/06/2015 01:22 PM - Eric Faulhaber

The types of failures I'm seeing:

- A buffer which previously was a top-level instance variable is referenced in an anonymous subclass of WhereExpression, stored in another top-level instance variable. Now that the buffer is scoped within a method (converted internal procedure), the compiler can't see it when processing the WhereExpression subclass. The solution is to pass the buffer reference to WhereExpression.evaluate. This problem most likely can occur with other inner classes (e.g., validation expressions) as well. This is not so much a regression, since the current conversion is broken: although the result compiles, it is not correct, since the record buffer must be scoped to the method, not to the top-level class.
- Cross-namespace conflict: a variable with the same name as a buffer was not renamed properly. This seems to be a true regression.
- The default buffer for a table is used in one internal procedure, and an explicitly defined buffer with the same name and for the same table is used in another. With my update, the buffer declaration/definition is moved to the converted internal procedure and the reference to the default buffer is lost. The existing conversion is wrong, too, however the result compiles: the same buffer is used as both the default buffer and the explicitly defined buffer scoped to the internal procedure. It is declared and initialized to null as an instance variable of the top-level class and only is initialized in the converted internal procedure where the explicit buffer was defined in the original 4GL. That means that with the existing conversion, we would have an NPE if we ever executed the method which references the default buffer before the method which defines the buffer. OTOH, if the methods were executed in the opposite order, the method which expected the default buffer would be using the wrong instance. With my update, it doesn't compile.

#7 - 05/06/2015 01:25 PM - Greg Shah

> With my update, it doesn't compile.

Congrats! You made it more correct. :)

#8 - 05/07/2015 03:18 AM - Eric Faulhaber

The third item in note 6 was actually a more serious latent bug than I thought. Buffer name conflict resolution was getting the Java buffer names out of sync with the Progress buffers. In this case, we had two buffer scopes for the default buffer, and the conflict resolution algorithm was overriding the Java name for one and not the other, effectively making them reference different buffers in the Java code.

I am testing a fix for all 3 items now.

#9 - 05/09/2015 12:53 PM - Eric Faulhaber

- File ecf_upd20150508a.zip added

- Status changed from WIP to Closed

- % Done changed from 0 to 100

The attached fix resolves the original problem and the items in notes 6-8. It has passed regression testing and is committed to bzr rev. 10851.

#10 - 11/16/2016 12:06 PM - Greg Shah

- Target version changed from Milestone 11 to Cleanup and Stablization for Server Features

Files

ecf_upd20150505a.zip	15.2 KB	05/05/2015	Eric Faulhaber
ecf_upd20150506a.zip	28.5 KB	05/06/2015	Eric Faulhaber
ecf_upd20150508a.zip	188 KB	05/09/2015	Eric Faulhaber