CoCoA-5 - Feature #531

Package protected variables should know which package protected them

09 Apr 2014 15:46 - John Abbott

Status:	Closed	**Start date:**	09 Apr 2014
Priority:	Normal	**Due date:**	
Assignee:	Anna Maria Bigatti	**% Done:**	100%
Category:	Parser/Interpreter	**Estimated time:**	8.00 hours
Target version:	CoCoA-5.4.0	**Spent time:**	3.10 hours

Description

A package protected variable should "know" which package protected it (and this info should be indicated along with any error message about the protection).

(protectionReason not set)

2021-01 fixed using CheckProtection

Related issues:

Related to CoCoA-5 - Bug #94: Default reason for protected variables	**Closed**	**29 Feb 2012**
Related to CoCoA-5 - Feature #346: Describe should indicate package name	**Closed**	**29 Apr 2013**
Related to CoCoA-5 - Bug #102: Packages: should exported functions be automat...	**Closed**	**09 Mar 2012**
Related to CoCoA-5 - Feature #622: New function: RandomSubset	**Closed**	**12 Sep 2014**
Related to CoCoA-5 - Feature #1174: package of a function --> of an identifier!	**Closed**	**05 Apr 2018**

History

#1 - 02 Sep 2014 11:33 - John Abbott

- Target version changed from CoCoA-5.1.1 Seoul14 to CoCoA-5.?.?

- Estimated time set to 8.00 h

#2 - 24 Apr 2015 11:34 - John Abbott

I do find it annoying when CoCoA says that name XYZ is a package-exported variable but then does not tell me which package it came from. Admittedly it is not so hard to grep through the packages in the packages directory; however, it is possible to have packages in other places (I know because I'm writing a new one, and it sits in a development directory until it's finished).

#3 - 11 Mar 2016 10:34 - John Abbott

No doubt some change will be needed inside the source for CoCoAInterpreter. It will probably take a long time to find out what to change, and probably not so long to actually make the change.

Anna also wants user defined fns from packages to know which package they were defined in; I don't know if this is the same problem (or at least resolved by the same change to the CoCoAInterpreter sources).

#4 - 15 Jan 2021 08:17 - Anna Maria Bigatti

- Related to Feature #1174: package of a function --> of an identifier! added

#5 - 15 Jan 2021 08:18 - Anna Maria Bigatti

- Target version changed from CoCoA-5.?.? to CoCoA-5.4.2

I might be able to fix this, see #1174

#6 - 15 Jan 2021 10:30 - Anna Maria Bigatti

- Status changed from New to Feedback

- Assignee set to Anna Maria Bigatti

- Target version changed from CoCoA-5.4.2 to CoCoA-5.4.0

- % Done changed from 0 to 90

Implemented: updated code of CheckProtection (extra arg runtimeEnv) in Interpreter.C
(done it, before I forgot the existance of $FindOwnerPkg$)

```
/**/ starting := 3;
--> ERROR: Cannot set "starting" (package-exported variable: $coclib)
--> starting := 3;
--> ^^^^^^^^
```

There is something called protectionReason but seems unset (very strange: the code for setting it looks good)

#7 - 15 Jan 2021 10:39 - Anna Maria Bigatti

modified also implementation of unprotect

```
/**/ unprotect starting;
--> ERROR: Cannot unprotect "starting" (package-exported variable: $coclib)
--> unprotect starting;
-->           ^^^^^^^^
```

#8 - 15 Jan 2021 12:23 - Anna Maria Bigatti

Anna Maria Bigatti wrote:

> There is something called protectionReason but seems unset (very strange: the code for setting it looks good)

protectionReason
it is here: void PackageStatement::implExecute, in Interpreter.C

Seems to be applied to the package itself and not to the variables of the package.

#9 - 15 Jan 2021 12:29 - Anna Maria Bigatti

- Description updated

#10 - 15 Jan 2021 20:42 - John Abbott

In the source code indicated by Anna, there is a loop which sets protectionReason (or some similar name).
These reasons are applied the variables which belong to the package *i.e.* whose names are of the form $pkg.var_name.
For some reason no such "reason" is given to the protected top-level variables which are set by the export commands.

Maybe investigate and fix?

NOTE: Anna's new function **PackageOf** makes this less urgent, but it would be nice if the error mesg about a protected top-level variable gives a helpful reason.

#11 - 22 Jan 2021 08:44 - Anna Maria Bigatti

- Description updated

All done: works for :=, unprotect and define.

The field protectionReason is actually set, but for the complete name (e.g. $anna.pippo), and not for the exported name. This could be fixed, but the current solution anyway works and applies to more cases.

#12 - 22 Jan 2021 09:52 - John Abbott

There is a sort of redundancy in package $anna putting a protection reason on the variable $anna.var.

It would be nice (and less redundant) if the code could be modified to protect also exported names with the correct reason.
Still, as I wrote in comment 10, this is now less pressing since PackageOf gives the same info (in a possibly nicer way?)

#13 - 14 May 2021 17:37 - John Abbott

- Status changed from Feedback to Closed

- % Done changed from 90 to 100