Summary on Local Variable Scoping

Article
Robert Cemper · Nov 11, 2017 3m read

Summary on Local Variable Scoping

This should be an overview over a subject that pops up over several places in online documentation mostly as remarks and never as dedicated chapter.

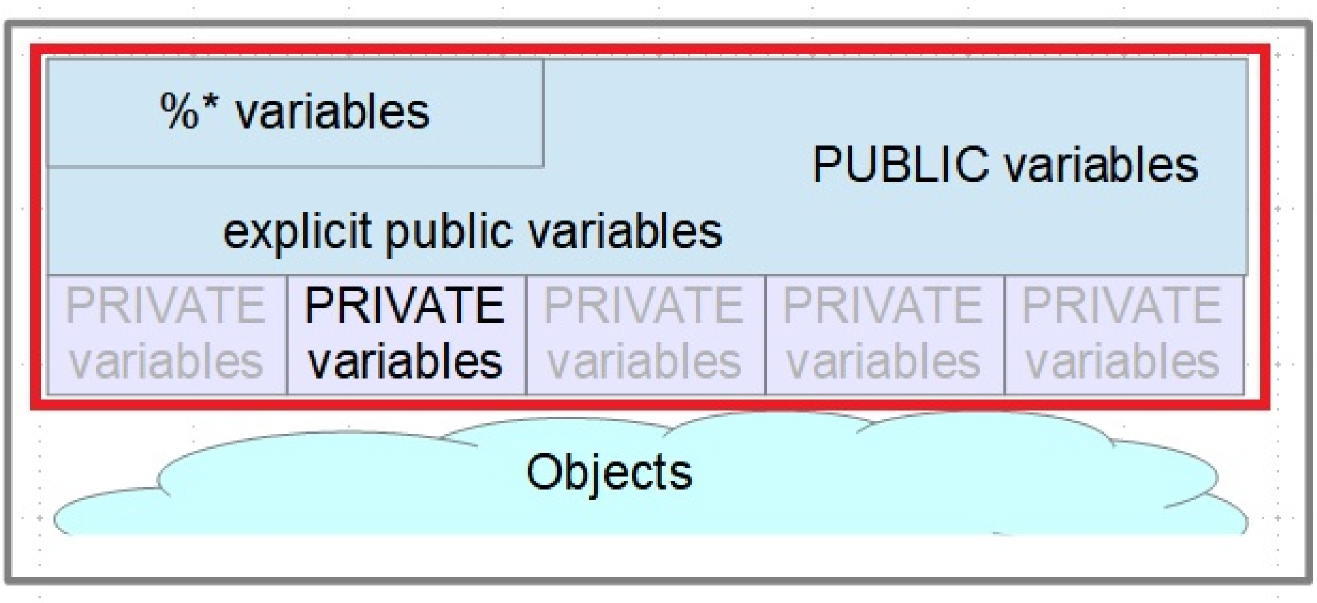
Once upon a time ... No it's not a fairy tale.
In the beginning of Caché (and before) you had your partition to run your code. Part of that partition was a space with all your local variables nicely sorted by %,A,..Z,a,...z

And whatever values or information you had to store locally was there and visible and available to any piece of code running in your partition. No problem if you are team of developers working well together with complete documentation and excellent discipline.
[sorry it shouldn't be a fairy tale].

If fact working with (own or foreign) software packages it could become a nightmare then and finding non conflicting use of variables was more effort than the code itself. Needless to mention that meaningful naming became an exception. The only help at that time was
the **NEW** command to push variables on stack and recover them later.
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOS_cnew

The real solution in Caché 5.0 (2002) was to create an isolated scope for procedures and methods with an empty and independent space for variables dedicated to these BLOCK of code. No trouble anymore with protecting against misused or identic named variables.
For Class and Method definitions the parameter was named **ProcedurBlock** .
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=ROBJ_method_procedureblock
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GORIENT_class_method_keywords

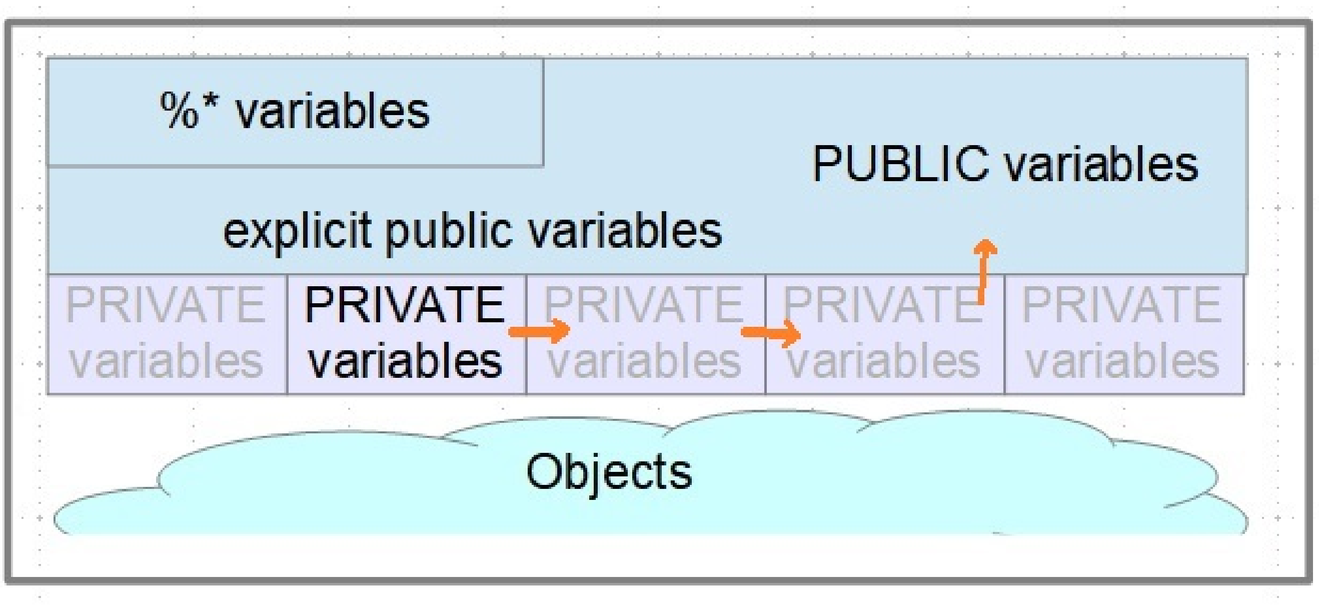
The traditional scope of variables was named **PUBLIC** and the new scope was name **PRIVATE**.
With the basic rule: anything starting with **%** is a PUBLIC variable. Anything else requires explicit setting.



The keyword ProcedureBlock (Default=1) controls if a PRIVATE scope for variables is available.
What a improvement!
Implementing easy to use functions, procedures became rather simple with scoping.
And implementing Methods in Object Classes became remarkable straight forward.
Now in parallel to all PUBLIC variables you could have a large set of PRIVATE variables.
(No excuse for od named variables anymore !)To make it clear: To the set of PRIVATE variables applies the Highlander Principle:
There can be only one ! http://wiki.c2.com/?HighlanderPrinciple

There is 1 set of Public variables and 1 set of Private variables active at a point in time.

But as in real life there is a minor exception: If you pass a variable *ByReference* to the called method / procedure you have a small door back to the calling scope.



The variant ProcedureBlock=0 or Not ProcedureBlock was mainly targeted to legacy code that was relying on old logic. BUT: every now and then there are some dark corners of generated code that still require / use it.

In general conflicts with variable scoping is a rare thing.
With 2 **exceptions** and 1 workaround:
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GCOS_usercode_indirxecjob

Xecute just works on public variables in its basic definition.
There is an extent that allows passing of parameters somewhat similar to a procedure call.
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOS_cxecute
A further extension in this direction is provided by the **$Xecute()** function.
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOS_fxecute

Indirection just works on public variables. No workarounds.
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GCOS_operators#GCOS_operators_indirection
Another good reason to stay away from indirection.

You find more related documentation on variable scoping here:
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GCOS_C16865
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GORIENT_ch_cos#GORIENT_cos_scope
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GCOS_usercode#GCOS_usercode_args_byref

#Beginner #Caché #InterSystems IRIS

140 2 1 3 554

Log in or sign up to continue
Add reply

Source URL: https://community.intersystems.com/post/summary-local-variable-scoping