
Question

[Laura Cavanaugh](#) · Nov 7, 2019

%OnNew and %OnAfterSave - do they user transaction processing under the covers?

Hello community!

I have a question about the %OnSave method of a class. We have a class that has two properties that are classes.

Class A

Class B

Class C

Class A.PropertyB as Class B

ClassA.PropertyC as Class C

Classes B and C also need to point back to Class A - it's just the way it is. We need to be able to use any one of these classes and get to the others (it's actually even more complicated than this, as Classes B and C also have a PropertyC and PropertyB as well, respectively) :

ClassB.PropertyA as Class A

ClassC.PropertyA as Class A

To see a simpler version of this question, see ** at bottom.

These two properties are created in the %OnNew for Class A

```
Method %OnNew() As %Status [ Private, ServerOnly = 1 ]
{
set ..PropertyB = ##class(Class.B).%New()
Set ..PropertyC = ##class(Class.C).%New()
Quit $$$OK
}
```

WHO KNOWS how long later (although that doesn't really matter yet), Class A is saved, and of course, PropertyB and PropertyC are saved with it in the %OnAfterSave() -- and PropertyB.PropertyA is set to this ClassA object, and PropertyB.PropertyA is set to this ClassA object.

```
(ClassA)Method %OnAfterSave(insert As %Boolean) As %Status [ Private, ServerOnly = 1 ]
{
set ok=$$$OK
try {
if (insert) {

Set ..PropertyB.PropertyA = $this

Set ..PropertyC.PropertyA = $this

$$$ThrowOnError(..PropertyB.%Save())
$$$ThrowOnError(..PropertyC.%Save())
}
}
catch (ex) {
do ex.Log()
set ok=ex.AsStatus()
}
Quit ok
}
```

We used to create PropertiesB and C and then save them within a TSTART/TCOMMIT (in other code, not in the %OnAfterSave()), so that we could be sure that the creation of all of these properties either ALL happened or NONE happened.

A side benefit is that we always know that ClassA.PropertyB.%Id() == ClassA.%Id() -- because the Ids are created at the same time, or not at all -- which we are trying to make make use of, to make some looping code much faster. This is sort of the crux of this matter.

Another developer here asked if we can be sure that the %OnAfterSave uses transaction processing.

My question is (finally): is the %OnAfterSave the way I have written it using transaction processing under the covers? What if ..PropertyC.%Save() encounters an error, and causes the %OnAfterSave to return an error - is the ..PropertyB.%Save() rolled back? What about the ClassA.%Save() - is that rolled back too?

**I guess if you'd like to simplify this question, you can, thusly:

Does the %OnAfterSave() use transaction processing such that if an error status is returned, is the %Save() that originated the call rolled back and everything that just happened within it rolled back?

Thanks for the help,

Laura

[#Database Transaction Processing](#) [#Object Data Model](#) [#Caché](#)