Article
Danny Wijnschenk · Juil 19, 2022 4m de lecture

Attention à mélanger OO et SQL

Utiliser des méthodes avec syntax objet et SQL est l'une des caractéristiques les plus intéressantes dans Object Script. Mais dans un cas précis, ça m'a donné des résultats inattendus, donc j'ai essayé d'isoler le cas et le décrire ici.

Disons que vous devez écrire une méthode de classe qui met à jour une seule propriété. Habituellement, j'écrirais cela en utilisant SQL comme ceci :

```
 ClassMethod ActivateSQL(customerId) as %Status
{
 &sql(Update Test.Customer Set Active=1 Where ID=:customerId)
 If SQLCODE'=0 {
 Set exception = ##class(%Exception.SQL).CreateFromSQLCODE(SQLCODE, $Get(%msg))
 Quit exception.AsStatus()
 } Else {
 Quit $$$OK
 }
}
```

et appelez cette méthode de classe partout où j'en ai besoin dans mon application.

Mais si le code de l'application a ouvert l'instance lorsque cette méthode de classe est appelée, et qu'il effectue ensuite un %Save, il écrasera les mises à jour qui se sont produites dans la méthode de classe :

```
 Set objCust=##class(Test.Customer).%OpenId(id)
Do objCust.ActivateSQL(id)
Set objCust.Name = "something"
Set sc = objCust.%Save()
```

En changeant l'ordre des lignes, le problème serait résolu, mais vous devez être très prudent avec ce type de mélange :

```
Do ##class(Test.Customer).ActivateSQL(id)
Set objCust=##class(Test.Customer).%OpenId(id)
Set objCust.Name = "something"
Set sc = objCust.%Save()
```

Lorsque la méthode de classe serait écrite en utilisant la syntaxe OO comme ceci :

```
 ClassMethod ActivateOO(customerId) as %Status
{
  Set objCust = ##class(Test.Customer).%OpenId(customerId)
  Set objCust.Active = 1
  Quit objCust.%Save()
}
```

il n'y aurait pas de problème puisque l'instance ouverte dans le code appelant et l'instance ouverte dans la méthode de classe pointeraient vers la même instance en mémoire.
(Outre une pénalité de performances, car l'ouverture d'une instance avec de nombreuses propriétés pour mettre à

jour une seule propriété est plus lente qu'une mise à jour SQL)

Donc en conclusion : méfiez-vous des instances d'ouverture "trop longues" le long de votre code si vous utilisez également SQL.

J'ai joint la classe de test complète au cas où vous voudriez la voir par vous-même, appelez Do ##class(Test.Customer).Test(0) pour voir le code en utilisant uniquement OO, et .Test(1) en utilisant le SQL (et voir que la mise à jour SQL est écrasée)
Tous les commentaires sont appréciés !

```
Class Test.Customer Extends %Persistent
{

Property Name As %String;
Property Active As %Boolean;
ClassMethod ActivateSQL(customerId) As %Status
{
 #Dim exception

 &sql(Update Test.Customer Set Active=1 Where ID=:customerId)
 If SQLCODE'=0 {
  Set exception = ##class(%Exception.SQL).CreateFromSQLCODE(SQLCODE, $Get(%msg))
  Quit exception.AsStatus()
 }

 &sql(Select Name, Active Into :name, :active From Test.Customer Where ID
 = :customerId)
 Write !,"Result After SQL Update : ",!
 Write "Name   : ",name,!
 Write "Active : ",active,!!
 Quit
}

ClassMethod ActivateOO(customerId) As %Status
{
 #Dim objCust as Test.Customer
 #Dim sc as %Status
 Set objCust = ##class(Test.Customer).%OpenId(customerId)
 Set objCust.Active = 1
 Set sc = objCust.%Save()
 If sc'=$$$OK Quit sc
 &sql(Select Name, Active Into :name, :active From Test.Customer Where ID
 = :customerId)
 Write !,"Result After %Save : ",!
 Write "Name   : ",objCust.Name,!
 Write "Active : ",objCust.Active,!!
 Quit
}

ClassMethod Test(mode = 0)
{
 #Dim objCust as Test.Customer
 #Dim sc as %Status
 #Dim id as %Integer
 ;Create an instance and keep the id in memory
 Set objCust = ##class(Test.Customer).%New()
```

```
 Set objCust.Name = "Danny"
 Set sc = objCust.%Save() If sc'=1 Write "Could not save",!
 Set id = objCust.%Id()
 Kill objCust

 ;Open and display the created instance
 Set objCust=##class(Test.Customer).%OpenId(id)
 Write "Name   : ",objCust.Name,!
 Write "Active : ",objCust.Active,!

 ;Call a classmethod that updates the id with SQL or OO
 If mode=0 {
  Do objCust.ActivateOO(id)
 } else {
  Do objCust.ActivateSQL(id)
 }
 ;Change the instance (that is still in memory)
 Set objCust = ##class(Test.Customer).%OpenId(id)
 Set objCust.Name = objCust.Name_" - edited"
 Set sc = objCust.%Save() If sc'=1 Write "Could not save",!
 Write "Name   : ",objCust.Name,!
 Write "Active : ",objCust.Active,!
 ;the sql update in the classmethod is overwritten with the instance that was still i
n memory
 ;Open and display the created instance
 Kill objCust
 Set objCust = ##class(Test.Customer).%OpenId(id)
 Write "Name   : ",objCust.Name,!
 Write "Active : ",objCust.Active,!
}
}
```

#Instructions de codage #ObjectScript #Caché

URL de la source:https://fr.community.intersystems.com/post/attention-%C3%A0-m%C3%A9langer-oo-et-sql