Question
Nigel Salm · Mar 16, 2021

Calling class methods of a class in another Namespace

Hi

I have a class in several namespaces. This class is what I call a common component in that it is a class that performs a certain set of functions that are common to a number of Interfaces. In this example the class is essentially a Message Queue. When a Data Event is created in my source database I want to invoke the CreateMessage() class method in all of Interface namespaces.

The common component class has a name DFI.Common.Queue.ODSMessageQueue. I can inherit the class into one or more classes e.g. DFI.Common.Queue.Interface1MessageQueue, DFI.Common.Queue.Interface2MessageQueue.

In my Data Source Namespace I have a class that has the following properties:

- InterfaceName
- InterfaceNamespace
- MessageQueueClassName
- IsActive
- IsLiveProduction

So there is a record for each interface, the namespace it is running in and the MessageQueueClassName which is either DFI.Common.Queue.ODSMessageQueue or if I use inheritance then the class names will be the inherited class name in that Interface Namespace.

So when a data event occurs in my Data Source Namespace then I will create a message in every message queue. So I walk through each record in my list of interfaces and using the MessageQueueClassName I can do the following:

set tSC=$classmethod(MessageQueueClassName,"CreateMessage",Param1,Param2,...Param*N*)

So there are two possible approaches:

1) I put the DFI.Common.Queue.ODSMessageQueue class into the Data Source Namespace and then I create a class mapping into each Interface namespace but I do not create a global mapping. This will ensure that the code is available in the Interface Namespace but it will store the underlying global in the Interface Namespace.

2) If I use inherited class names that are specific to each Interface then the mapping is reversed. The class is mapped to the Data Source Namespace. If I map the global to the Data Source Namespace then as long as class names are different there is no issue. But if I have Interface QC Namespace and a Production Interface Namespace that use the same MessageQueueClassName then the data from the one namespace will be written into the same global as the second namespace. To compensate for this I could build the Primary Key/IDkey on the class as being The InterfaceName, Namespace, Generated Integer ID.

e.g. ^DFI.Common.Queue.Interface1MessageQueue({nterfaceName},{Namespace},RowId)={$lb(message properties})

I don't like this as I would rather have a system RowId so that I am not constrained from using Bitmap Indexing. I also have heve very long ID's in the form:

{InterfaceName}||{NameSpace}||{RowId}

So, at this point I have a couple of different approaches that use Class and Global Mapping.

So my question is:

From my Data Source Namespace can I call the methods in my Message Queue Classes which reside in each Interface Namespace without using mappings and without using namespace swapping. By that I mean:

set currentNS=$namespace

zn InterfaceNamespace

set tSC=##class(.....).CreateMessage(p1,p2,...,pN)

In the Interface Namespace I have a Business Service, DFI.Common.Service.MessageQueueService, which calls a method GetNextMessage()

The message class has the following groups of properties:

The Timestamps

CreateTS

ProcessTS

CompledTS

The Data Fields

These are the fields of data that I want to send to my Interface which will use the information in these fields to generate either an HL7 PIX/PDQ Message or a FHIR Resource JSON. These properties would be specific to ypur data source and the information that is required by the Interface.

The Status Fields

The Document Type,

Message Status,

HTTP/FILE/SQL Response Status

HTTP Response Body JSON or HL7

Human Readable Error Message

The Process

The GetNextMessage() finds the next message where ProcessTS and CompletedTS are NULL

Having found a message set the ProcessTS to %TimeStamp

Create a Request Message to send to the Business Process that has one property: MessageID

Process the message in the Business Process, access the required data, transform it into the target document type

Despatch to the Business Operation (HTTP, File (for testing purposes))

Receive a Response

Call the CompleteMessage() Method that updates the status fields

Job Done

Other Methods

Purge Message Queue older than N days

Resend Messages (sets the processTS and CompletedTS and Status fields to NULL, thereby restoring the message to unsent)

I give you thi imformation because I have a question about it in the Poll Section

Thanks

Nigel

#Ensemble
Product version: IRIS 2020.2
$ZV: IRIS for Windows (x86-64) 2020.1 (Build 217.1U) Wed May 27 2020 14:28:49 EDT

Source URL: https://community.intersystems.com/post/calling-class-methods-class-another-namespace