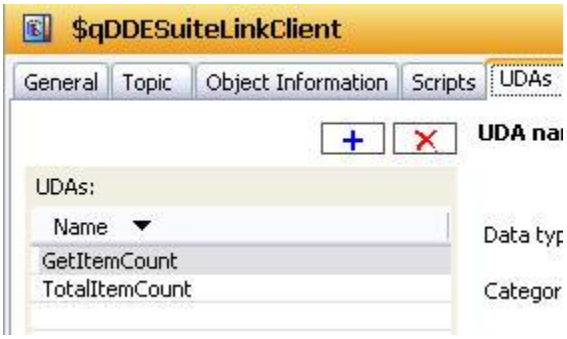
Introduction

When designing, implementing, or troubleshooting a system, it is always useful to have the total number of items being polled from a given DI Object. The total number of I/O being polled by a DI Object, if excessive, can lead to system instability/poor performance. As many implementations of System Platform, including those using the Base template library, utilize Dynamic Attribute Assignment at runtime in order to take full advantage of the object-oriented nature of the software, the total number of items being polled from a given DI Object can vary depending on what objects are deployed or on scan. Thus, the most reliable way to get the total number of items being polled from a DI Object is to look at the .ItemCnt attribute of each scan group on the DI Object. While this is easy to do for DI Objects with single topics, totaling up item counts of DI objects with multiple topics can be cumbersome. The following article describes steps to create a script which will automatically total up the number of item references on a given DI Object.

Automatically Calculating Total Number of Items on a DI Object

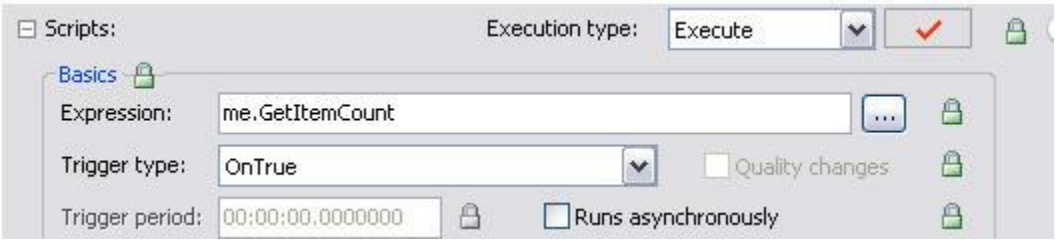
First, if not already done, create a derived template of the desired DI Object base template. Open this derived template. Next, navigate to the UDAs tab. Create two UDAs:

GetItemCount – Boolean / User Writeable
TotalItemCount – Integer/ User Writeable

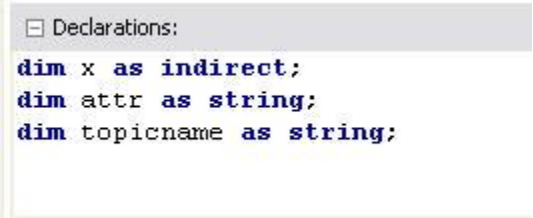


Next, navigate to the scripts tab. Create a new script with the following properties:

Script Name: GetItemCount_ot
Script Type: Execute
Script Trigger: me.GetItemCount
Trigger Type: On True



Now, Open the Declarations section and insert the following lines – these lines will define the indirect and string local variables used throughout the script.



Finally, add the body of the script below. This script uses the Scangrouplist string array attribute inherent in all DIObjects. In summary, the script loops through each element of the array, then uses the BindTo function of the indirect local variable to connect to that scan group's .ItemCnt attribute. Finally, it adds the value of the .ItemCnt to a variable collecting the total number of items. At the end, it assigns that value to the TotalItemsCnt UDA, and resets the flag.

```
'*********************
'Wonderware Northeast getItemCount script
'Script Type: Execute
'Script trigger: me.GetItemCount
'Trigger type: onTrue
'Variable requirements:
'    UDAs: GetItemCount (boolean), TotalItemCount (integer)
'    local variables in declarations: x (indirect), attr (string)
'Purpose: Cycle through .ItemCnt attributes of each scangroup in DI Object to get
'an accurate count of the items being requested from the DI Object.
'*********************
dim totalitems as integer;

'reset the flag
me.GetItemCount = 0;

'Reset the current Total Item Count to zero:

totalitems = 0;

'For each one of the topics:

For each topicname in me.ScangroupList[]
   'build the string for the .ItemCnt attribute
   attr = "me." + topicname + ".ItemCnt";
   'bind the indirect variable to the topic's .ItemCnt attribute
   x.BindTo(attr);
   'get the current value of the indirect, add to the total item count, then
continue loop
```

```
   totalitems = totalitems + x;
Next;

'assign new total items to attribute
me.TotalItemCount = totalitems;
```

After adding, create your instances and templates. Then, to get the current item counts, set the GetItemCount UDA of the DI Object to true. In the example below, the myDIObject object has three scan groups associated with it, named TestAreaA, TestAreaB, TestAreaC. The TotalItemCount shows the total of those three scan groups.

AttributeReference	Value	Quality	Status
myDIObject.TestAreaA.ItemCnt	30	C0:Good	Ok
myDIObject.TestAreaB.ItemCnt	15	C0:Good	Ok
myDIObject.TestAreaC.ItemCnt	40	C0:Good	Ok
myDIObject.TotalItemCount	85	C0:Good	Ok
myDIObject.GetItemCount	false	C0:Good	Ok