



University of Colorado

Boulder | Colorado Springs | Denver | Anschutz Medical Campus



## OnBase Guide - WorkView for Unity Form Reporting

---

**Goal:** To report on non-keyword fields on Unity forms using Reporting Dashboards via WorkView

**Complexity Level:** Departmental WorkView  
Administrative Users

6/3/2020

## Table of Contents

Background .....	3
Pre-Requisites .....	3
Steps to Complete in WorkView (Studio) .....	3
Steps to Complete in Workflow (Studio) .....	6
Module Association (Studio) .....	11
Steps to Complete in Reporting Dashboard Configuration (Unity client) .....	11
Testing the Solution .....	12
Migration between Environments .....	12
Deleting Objects Upon Form Deletion .....	12



---

## Background

Reporting Dashboards can be configured to display keyword data for OnBase documents, but non-keyword (XML) Unity form fields are not available for direct reporting using this Document Query data provider method. However, using workflow to create WorkView objects with the information from the form allows for reporting on the WorkView objects to display this information.

As this is a complex process, please consider whether it is better to store the form values as keyword values in order to report on keyword values directly. In most cases it is recommended to use keyword values, but in some cases (like fields containing more than 250 characters) creating WorkView objects will be a better option.

---

## Pre-Requisites

You will need to have Studio installed, refer to the [OnBase Studio - Installation and Setup](#) guide if needed.

You'll also need to be comfortable with Reporting Dashboard, Workflow and WorkView configuration and have the necessary administrative user groups assigned to your account. Refer to the [WorkView Configuration Guide](#) and the Reporting Dashboards, Workflow and WorkView MRGs for more detailed information than what is included in this guide.

The Unity form will need to be configured. Make sure the fields have been given a descriptive ID so you can easily identify the desired fields. Refer to the Unity Form MRG for more information if needed.

Once the form is configured and your reporting needs are defined, the WorkView application can be planned and configured. Then the workflow can be configured to support the process of creating objects. Finally, you can configure Module Associations and the dashboard.

Contact [UIS\\_DM\\_Support@cu.edu](mailto:UIS_DM_Support@cu.edu) for assistance if needed.

---

## Steps to Complete in WorkView (Studio)

A WorkView application will need to be created in order to store the data from the Unity forms as WorkView objects. Any data that you want included in the dashboard will need to be attributes on a class in the application. Then filters for the classes will be the basis for the Module Associations to use the objects in reporting dashboards.

For example, we want to report on the non-keyword values in the multi-line textbox, radio button and checkbox on this form template, in addition to the student keyword information.



# Form for WorkView Reporting

<b>Student ID*</b>	<b>First Name</b>	<b>Last Name</b>
<input type="text"/>	<input type="text"/>	<input type="text"/>
<b>Date of Birth</b>	<b>Campus Code*</b>	
<input type="text"/>	<input type="text" value="CUSYS"/>	
<b>Textbox</b>		
<input type="text"/>		
<b>Question</b>		
<input type="radio"/> Yes <input type="radio"/> No <input type="checkbox"/> Option		
<b>Submit</b>		

This will be done using one class in an application created for this purpose. Each form field that will be included in the dashboard has a corresponding attribute in the class.

Repositories

DMOTST - Larissa Armand - CUSYS OnBase Admin  
<https://dm-tstapp.qa.cu.edu/AppServer/service.aspx>

Integration Services  
Workflow

Find
 

- B - GSS - Graduate Faculty Appointments (LA)
- C - BUR - Customer Service Data
- C - HR - Case Management
- S - ES - Physical Records Management
- S - UIS - Unity Form Reporting Example
  - Classes
    - SUISUnityFormValues (S - UIS - Unity Form Values)
      - Mappings
      - Filters
      - Folders
      - Actions
      - Views
      - Screens
      - Triggers
      - Constraint Sets
      - Notifications
      - Filter Bars
      - Mobile Apps

Properties

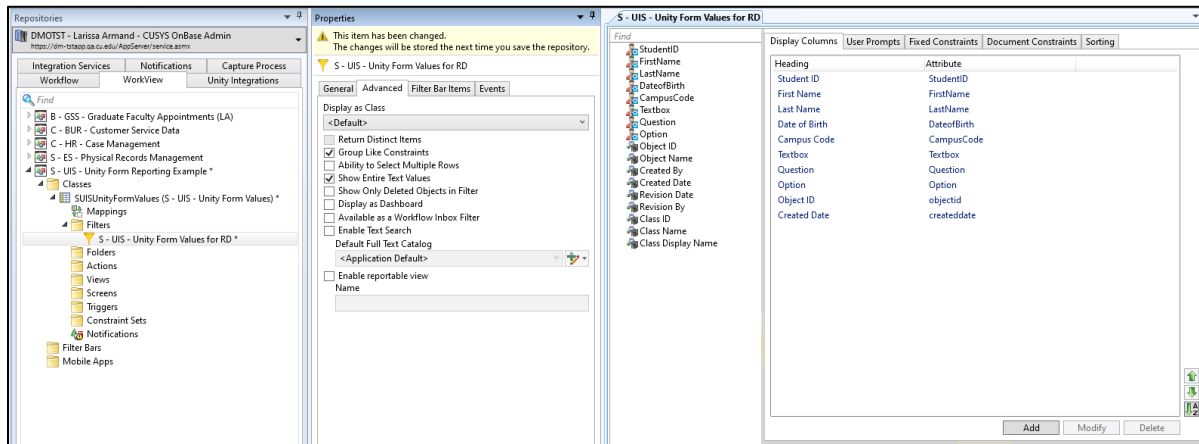
SUISUnityFormValues

AttributesGeneralEventsAdvancedScriptsUser Groups

ID	Name	Display Name	Data Type	Index
1122	StudentID	Student ID	Alphanumeric (9)	None
1123	FirstName	First Name	Alphanumeric (50)	None
1124	LastName	Last Name	Alphanumeric (50)	None
1125	DateOfBirth	Date of Birth	Date	None
1126	CampusCode	Campus Code	Alphanumeric (5)	None
1127	Textbox	Textbox	Alphanumeric (500)	None
1128	Question	Question	Alphanumeric (3)	None
1129	Option	Option	Boolean	None

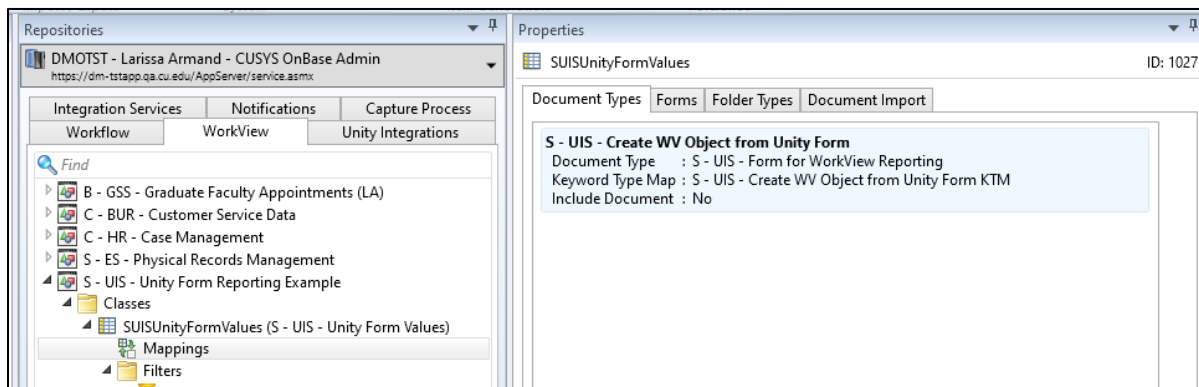
NOTE: It's recommended to disable direct creation of objects in the class(es) and set user group access accordingly if objects should only be created from Unity forms through the workflow process.

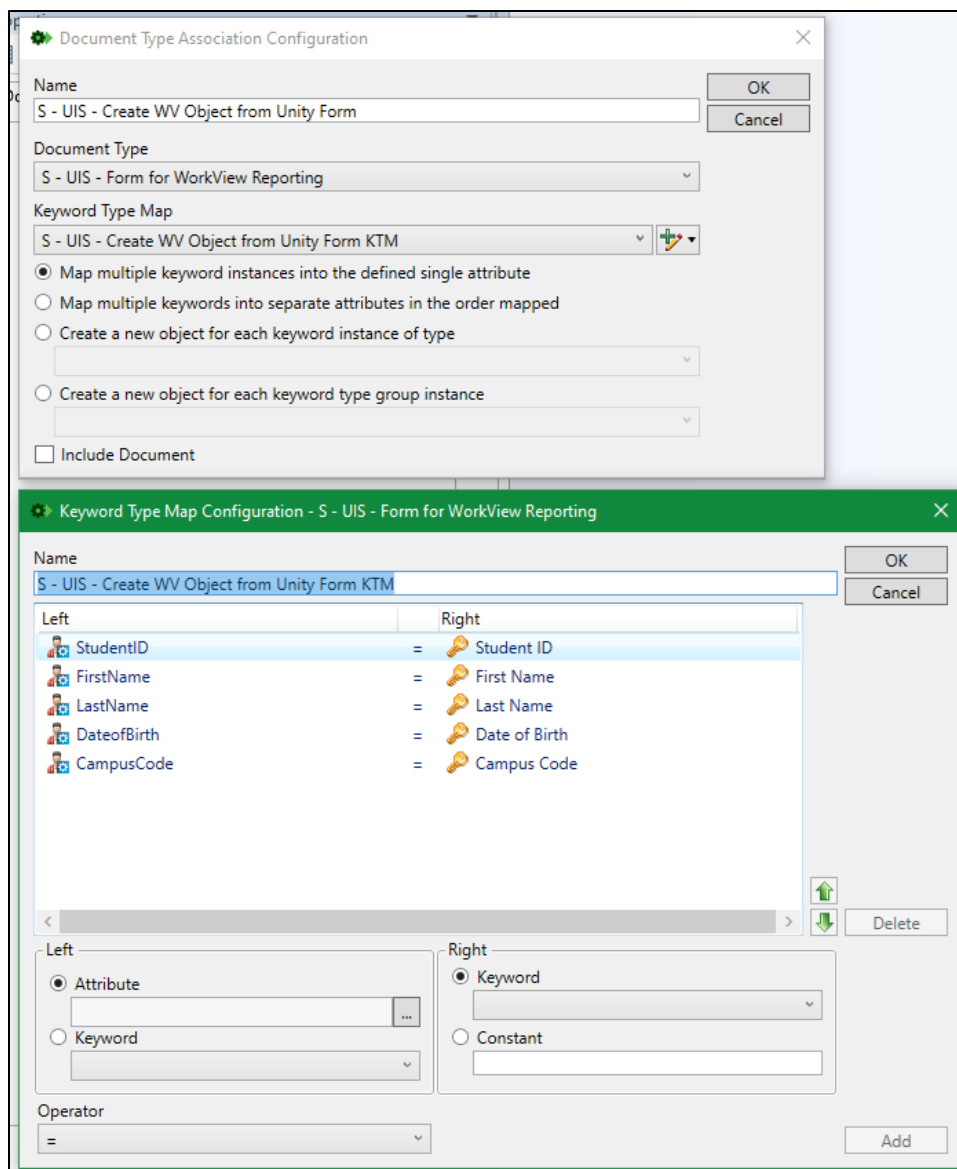
A filter also needs to be created to display all the class attributes, plus some object metadata attributes are included. All attributes that should be in the dashboard need to be included in a filter.



NOTE: In order to view all objects directly as an administrator, you'll also need to create a filter bar and assign the filter and your administrative "WorkView" user group.

Finally, a Document Type Association is needed in order to map the document keyword values to the WorkView attributes. This is used in the workflow action to create objects.





## Steps to Complete in Workflow (Studio)

For this process, the Unity forms will need to be directed to a Workflow life cycle which creates WorkView objects. The life cycle will need to be a Unity life cycle to create the relationship between the form and a WorkView object. You could:

- Create a life cycle for the appropriate document type to create objects upon form submission
- Create a life cycle and direct forms there from another life cycle after they are processed
- Add a queue or actions to an existing Unity life cycle

In this example, the forms will be directed to a life cycle upon submission and the life cycle will only create WorkView objects.



Add an action with the action type “Create WorkView Object from this Document.” Choose the document type and document type association you configured for the application. Check the box to save the object ID to a property. This will be needed to copy the non-keyword values from the document to the object.

The screenshot shows the 'Create WorkView Object' configuration window. At the top, it displays the title 'Create WorkView Object' and the ID '25130'. Below this, it shows the creation and modification dates and times, both set to 12/10/2019 12:16:00, by 'Larissa Armand - CUSYS OnBase Admin'. The 'Action Type' is set to 'Create WorkView Object from this Document'. The 'General' tab is selected, and the 'Documentation' sub-tab is active. Under 'Document Type Associations', there is a table with two rows: 'Document Type' and 'DTA'. The first row shows 'S - UIS - Form for WorkView Reporting' and 'S - UIS - Create WV Object from Unity Form'. Below this, there is a checkbox for 'Display object' and a 'Remove' button. The 'Document Type' is set to 'S - UIS - Form for WorkView Reporting'. The 'Document Type Association' is set to 'S - UIS - Create WV Object from Unity Form (Class SUIUnityFormValues)'. There is an 'Add' button next to it. The 'Save Object ID to Property' checkbox is checked, and the property name 'propObjectID' is entered in the text field. At the bottom, there is a 'Use Scoped Property Bag' dropdown menu and three checkboxes: 'Disable', 'Enable Debug Breakpoint', and 'Log Execution'.

For any non-keyword values we want to include, we'll need to copy those values from the form to a property. For example, we do the following for the textbox on the example form.



Action Type  
Copy Property to/from Form Field

General Documentation

Property Name  
propTextbox

Form Type  
S - UIS - Form for WorkView Reporting

Form Field Name  
multilinetextbox

☐ Copy To Form Field  
☒ Copy From Form Field

Use Scoped Property Bag

☐ Disable ☐ Enable Debug Breakpoint ☐ Log Execution

We then need to identify the related WorkView object to copy these properties to attributes on the object. We do so with a “Related Item Exists” rule.

Rule Type  
Related Item Exists

General Related Documentation

Target  
Related Item

**Located By** Ad Hoc Portfolio Relation

Source Content Type  
- Documents

Related Content Type  
- WorkView Objects

☒ Use Related Items For Tasks  
☐ Allow User To Choose Items For Task Execution

Evaluate to true when

Operator Type  
>

☒ Constant value  
 0

☐ Property

Use Scoped Property Bag


☐ Disable ☐ Enable Debug Breakpoint ☐ Log Execution

Then configure the relationship to map the object’s object ID to the object ID we stored as a property when creating the object. In this example, we’re using an ad hoc portfolio relation between the document and a WorkView object.






What type of related items will this relation find?

Content Type 

Documents ▼

Related Content Type 

WorkView Objects ▼

A custom keyword type mapping was used to map the object's object ID with the object ID stored as a property.

What are the custom keyword type mappings?

Application

S - UIS - Unity Form Reporting Example ▼

Class

SUISUnityFormValues ▼

Attribute	Oper...	Type	Value
objectid	=	Property (Scoped)	propObjectID

Add Modify Delete

On the rule, select the checkbox to “Use Related Items for Tasks” then when create a “Set Attribute Value(s)” action for when the rule evaluates to true. This will set attribute values from the properties copied from the form.



Action Type  
Set Attribute Value(s) ?

General Documentation

Application  
S - UIS - Unity Form Reporting Example

Class  
SUISUnityFormValues

Attribute Values

Name	Type	Value
Question	Property	propQuestion
Option	Property	propOption
Textbox	Property	propTextbox

Remove

Attribute:

Constant Value

Property

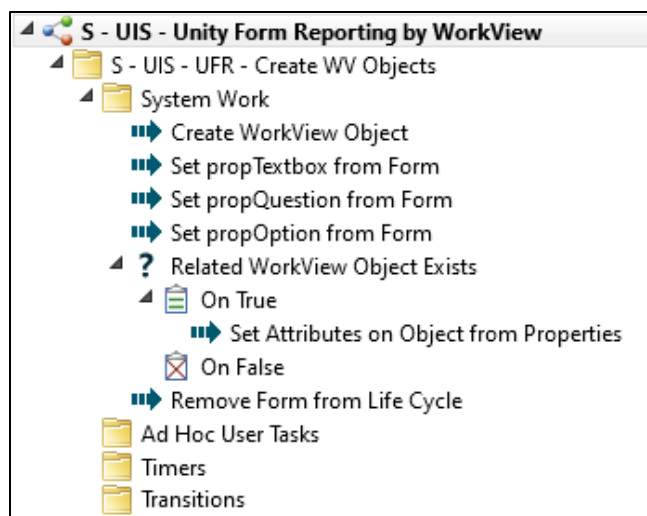
Add

Use Scoped Property Bag

☐ Disable ☐ Enable Debug Breakpoint ☐ Log Execution

After this step, the form is removed from our example life cycle.

Overall, the workflow looks like this:



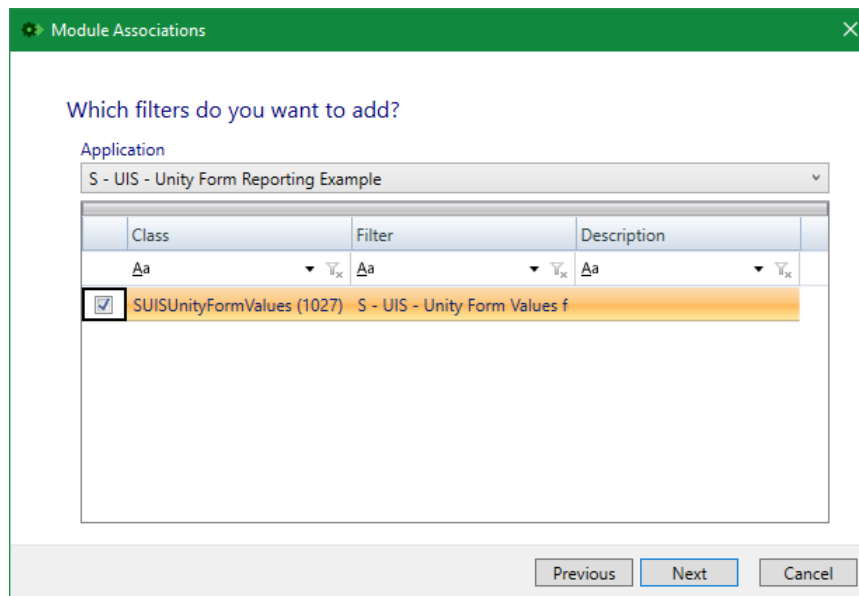
For production processing, it's recommended that initial system work be configured on a timer. This example has been simplified for clarity.

---

## Module Association (Studio)

Module Associations map filters from WorkView applications so that they can be accessed by data providers. Instructions for creating Module Associations are available in the WorkView MRG and in Section 2 of the [Reporting Dashboards Beyond the Basics course](#) on Premium.

On the WorkView tab in Studio, select **Module Associations** then click **Add** to create new. Choose **Reporting Dashboards**. Select the application and filter. Choose the desired user groups and finish.



---

## Steps to Complete in Reporting Dashboard Configuration (Unity client)

Instructions for creating a data provider from a WorkView module association and a dashboard using the data provider are available in the Reporting Dashboards MRG.

Create a new data provider, choose WorkView as the data provider type, then choose the desired Module Association. Keep in mind that the app pool will need to be recycled before the module association(s) will appear in the Unity client.

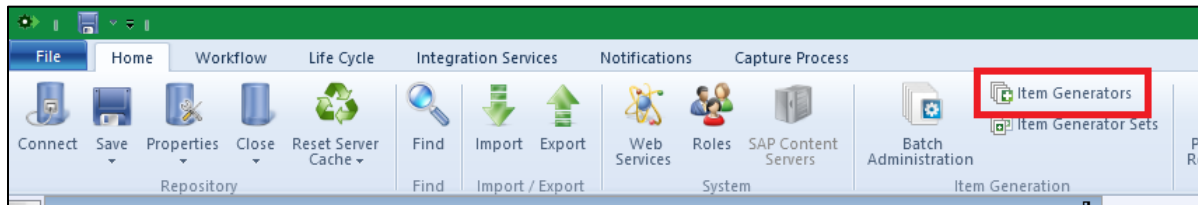
Once the data provider is configured, the rest of the dashboard configuration will be the same as if another data provider type had been used.

NOTE: Users who will access the dashboard will need to be granted access to the dashboard, data provider(s) and module association(s) involved.

---

## Testing the Solution

Item Generators can be configured to create Unity forms in the desired document types and with the desired keyword/field values. These let you test life cycles in a variety of ways and specify the starting queue and entry date. Refer to the Studio MRG for more information on Item Generators.



Feel free to create your own Item Generators to suit your needs with your own document types, life cycles, queues and values.

---

## Migration between Environments

There are special considerations for migrating projects involving WorkView, especially the reporting elements. The [Exporting and Importing guide](#) has further details.

---

## Deleting Objects Upon Form Deletion

If you'd like to have objects that were created through this process deleted when the source form is deleted, add the following:

- A field on the form to hold the Object ID for the object related to the form.
- An action to copy the object ID property to that form field.

A screenshot of the 'Action Type' configuration window. The 'Action Type' dropdown is set to 'Copy Property to/from Form Field'. Below this, there are two tabs: 'General' and 'Documentation'. The 'General' tab is active. It contains the following fields: 'Property Name' with the value 'propObjectID', 'Form Type' with the value 'S - UIS - Form for WorkView Reporting', and 'Form Field Name' with the value 'objectID'. At the bottom, there are two radio buttons: 'Copy To Form Field' (which is selected) and 'Copy From Form Field'.

- A Unity system task that will delete the form and the related WorkView object. This will require that the system task is used to delete the form rather than the standard document delete function (document type overrides on the relevant user groups can enforce this).

- Get the objectID property from the form.
- Identify the related WorkView object based on the property value. Use the same relationship as described above for copying property values to the object and select “Use Related Item for Tasks.”

Rule Type  
Related Item Exists

General Related Documentation

Target  
Related Item

**Located By** Ad Hoc Portfolio Relation

Source Content Type  
- Documents

Related Content Type  
- WorkView Objects

☒ Use Related Items For Tasks  
☐ Allow User To Choose Items For Task Execution

Evaluate to true when

Operator Type  
>

☒ Constant value  
0

☐ Property

- Delete the object.

Action Type  
Delete Object

General Documentation

☒ Current Item  
☐ Get Object ID from Property

From Specific Class (optional)

Application

Class

Selecting a specific class to help identify the object is optional but safer.

Clear

- Delete the form.
- Overall, the system task looks like this:



