OnBase Guide - Unity Script - How to use the “GEN - Read Environment Variable” Script

**Goal:** To use the “GEN - Read Environment Variable” script in Workflow to perform environment-specific processing based on the result.

**Complexity Level:** Departmental Workflow Developers

3/21/2019
Table of Contents

Background .................................................................................................................. 3
Prerequisites ............................................................................................................... 3
Set Property Values Needed for the Script ................................................................. 3
Run the Script .............................................................................................................. 6
Use the Script Results ................................................................................................. 7
Troubleshooting ......................................................................................................... 9
Background
This script will read an environment variable from the OnBase environment such that you can create environment specific workflow actions. Each OnBase environment will have one of these designations: DMOPRD, DMOUSTG, DMOTST and DMODEV.

Use Cases:
- Save files off to different UNC paths or NFS locations depending on the OnBase environment.
- Read files from different UNC paths or NFS locations depending on the OnBase environment.
- Call different web services domains depending on the OnBase environment.
- Other environment specific tasks.

When setting any property values to run a script or use script results, be sure to note which property bag is being used and be consistent with this selection.

Prerequisites
You must have OnBase Studio installed and know how to configure a life cycle. Refer to the OnBase Client Guides for instructions on installation and to the Workflow MRG for more details as necessary. Contact UIS_DM_Support@cu.edu for assistance if needed.

Set Property Values Needed for the Script
Create an action and choose the Set Multiple Property Values action type.

Click Add at the bottom of the panel at the right side to add a new property value.
Enter the following values to set the **UnityScriptResult** property:
- **Property Name:** UnityScriptResult
- **Constant Value:** Initialized (or leave blank)

Then click **OK**.

Click **Add** at the bottom of the panel at the right side to add a new property value and enter the following values to set the **ScriptError** property:
- **Property Name:** ScriptError
- **Constant Value:** [leave blank]
Then click **OK**.

Click **Add** at the bottom of the panel at the right side to add a new property value and enter the following values to set the **EnvironmentProperty_Name** property:

- Property Name: EnvironmentProperty_Name
- Constant Value: [leave blank]
Then click **OK**. Overall, the Set Multiple Property Values action should look like this when complete:

![Set Multiple Property Values dialog](image)

**Run the Script**

Create a “Run Unity Script” action. Select “GEN – Read Environment Variable” from the list of available scripts. Check the box to Refresh item after script has executed.
When the script runs, it will update the EnvironmentPropertyName property value with the current environment name.

**Use the Script Results**

In order to specify what processing should be done for each environment, you will need to create a rule to check for each possible value of EnvironmentPropertyName and list the actions to be performed for each environment.

Create a “Check Property Value” rule using the following values:

- **Property Name:** EnvironmentPropertyName
- **Operator Type:** =
- **Constant Value:** DMOPRD
Add the desired actions to perform if the environment is DMOPRD to the “On True” section of the rule.

Create “Check Property Value” rules for the other environments and add the desired actions to the rules as previously described. Overall, your configuration should look something like this:
**Troubleshooting**

If you encounter an issue, checking the values of the UnityScriptResult and ScriptError properties may provide helpful information for determining the cause of the issue.

While troubleshooting, it may be helpful to add a note to the document after the script is run in order to view the UnityScriptResult and ScriptError values.

![Table of Action Type, Create Note, Target, Note Contents, and Keyword Type]