Building ServiceNow Workflows

Build Service Catalogue

From the Filter Navigator bar found on the upper left hand side of your ServiceNow page, type in "Service Catalog" and select the option for "Catalogs"

Creating a Network Automation Catalog

To create a catalog that will encompass all Network Automation efforts, click on the blue "New" button and assign the parameters necessary.

Make sure that the "Active" checkbox is marked, and the application's scope is set to "Global".

All of our Catalog Items will reference this Catalog, so be descriptive within its name to make it easier to remember in subsequent steps.

Service Catalogue Category

We will need to build a set of "Categories" to allow us to group common workflows, such as "Wireless Automation" or "SD-WAN Automation". These groupings will be listed under the Catalogue created in our previous step.

From the Filter Navigator bar found on the upper left hand side of your ServiceNow page, type in "Service Catalog" and select the option for "Categories".

Selecting the blue "New"button towards the top of the page, define the necessary parameters such as description and image. It's important to make sure that the application's scope is "Global" and the "Parent" field references the Catalogue created in our initial state. This association builds the relationship between the Category and Catalogue.

Catalog Items

For the actual Service Request item, we'll need to revisit the search bar found on the upper left hand corner of our page and type in "Service Catalog", this time selecting "Items".

After clicking the blue "New" button at the top of your page, begin filling in basic parameters of your Service Catalogue Item. It is critical that you create the association to your Catalogue and Category.

Should your workflow require form fields to be filled by the user, then make sure to visit the "Variables" section towards the bottom of the page and select the "New" button.

Variables

Again, critical that the "Catalog Item" references the item created in our previous step. The "Type" dropdown will allow you to craft what type of Form Field should be presented to the user. A commonly used option in my example was "Single Line Text".

Selecting the "Mandatory" and "Active" checkboxes are self-explanatory, but the "Order" field needs to be considered if you're planning on presenting the Questions in a structured order-of-operations; a lower "Order" will move the question towards the top of the form.

A quick overview of the options available for you under the Question box:

- "Question" form field will be what is prompted to the user
- "Name" field will be the actual variable name (very important for our workflow process later on)
- "Tooltip" will be what the user sees when they hover their mouse of the field
- "Example Text" will be optional text provided within the field to help guid the user.

Process Engine

In the middle of our Catalog Item page, we see a box with several tabs. "Process Engine" will be the option that allows us to tie our Service Catalog to a workflow.

We will cover this phase in our next section, but note that you'll need to revisit this section of our Catalog Item very soon.

Workflow Editor

To build a workflow for execution, revisit the search bar on our left hand corner and type in "Workflow Editor". From here we will begin to build our first workflow and associate it to the Service Catalog.

Creating a workflow

Clicking the blue "New Workflow" button towards the top of our screen, we will be given a pop-up menu to fill in before we can start working on our blank canvas.

Assign a name to the appropriate field, and under "Table" select "Requested Item[sc_req_item]". Selecting the "Submit" button will present your workflow Canvas.

Creating the pipeline

Your canvas will have a "Begin" and "End" phase already defined, it's up to us to create everything in between. Right click on the line that connects "Begin" and "End" and select the appropriate task. Commonly used fields are:

- "Approval - User": require approval from a teammate before executing the task
- "Run Script": allows us to tie in an automation task

Approval - User

Double clicking on the workflow item will allow us to define the appropriate parameters, such as teammates or group approval requirements.

Run Script

This will be our mechanism to associate our workflow to an Automation task. We have yet to define the Automation task, but let's go ahead and start working on some boilerplate code.

```
try {
    var r = new sn_ws.RESTMessageV2("Ansible_Tower_Azure",
"change_psk");

// SITE SECTION
    r.setStringParameter("snow_ssid_id", current.variables.snow_ssid_id.
toString());
    r.setStringParameter("snow_ssid_psk", current.variables.
snow_ssid_psk.toString());

    var response = r.execute();
    var responseBody = response.getBody();
    var httpStatus = response.getStatusCode();
}

catch(ex) { var message = ex.message; }
```

RESTMessageV2

This is the preferred mechanism to execute a REST API call to a remote system, such as Mist or CSO. Make sure to pass in the REST Message name, followed by the Message Type (both have yet to be defined).

Variable assignment

Here we will insert the Catalog Item's variables, creating a new variable object by converting the value entered by the user into a string object.

This new variable object will be passed into the REST API call, tying the input from our user's form fields to the body of our message.

Execution

Make sure to include the section of code that tells SNOW to execute the function, store the output and status code to new variables.

Publish Workflow

Before moving on, make sure to click the three ticked "Hamburger" menu and set the Workflow's status to "Published"

REST Messages

For the REST API call definitions, start again with the search bar on your left hand corner of the screen and type in "REST Messages".

Selecting the blue "New" button towards the top, we'll begin defining basic parameters of our API messages.

HTTP Methods

This will define how we interact with the API endpoint.

Name

This field will be used to reference back to our Workflow, where we had defined the REST Message and Method name.

HTTP Method

This dropdown menu will enable us to select with API method we'll be using, most likely will be a POST operation.

Endpoint

This will be our API URI endpoint for the Ansible Tower server or Mist / CSO API services.

```
http://ansible-tower.centralus.cloudapp.azure.com/api/v2/job_templates
/36/launch/
```

Authentication Type

Depending on the API endpoint, this will help define the authentication system on the remote system. Reach out to the appropriate development team to understand which API authentication system is required.

Basic Auth Profile

This is filled in when using a combination of USERNAME/PASSWORD for API authentication. Create a profile where the appropriate fields are filled in, and select it within our REST Message search field.

HTTP Request

This section enables the users to enter fields found within the API request, such as HTTP Headers and the message body.

Headers

Under HTTP Headers, click on the field "Insert a new row..." and type in `Content-Type` and set the value to `application/json`.

Body

Under the "Content" window, enter the body of the API call in JSON format.

```
{
    "extra_vars": {
        "site_name": "${site_name}"
    }
}
```

This is where we pull in the variable created in our RESTMessageV2 javascript body, and fill it into our body with a format of `${variable_name_goes_here}`

Variable Substitution

Here we build the association of our variable created in the Workflow to a new variable for our REST message body.

Stitching everything together

Catalog Item

Validate the following:

- the Catalog Item is associated to the appropriate Catalog Category
- the "Process Engine" section references the Workflow by name
- the variable names created here are inline with those called in the RESTMessageV2 script.

Workflow

Validate the following:

- variables called from the Catalog Item are correct
- new objects created based on the form fields are passed to the REST Message
- the RESTMessageV2 references the appropriate name for our REST Message and associated METHOD

REST Message

Validate the following:

- HTTP method aligns with the API URI endpoint's allowed methods
- Variable Substitution section is referencing the correct variable names defined in the RESTMessageV2 script of our Workflow
- Authentication is setup appropriately

A "Test Run" can be issued to validate the success of your API call.

Wrapping up

There is a lot of "leg bone is connected to the hip bone" games played to stich the various components together. Should you run into any issues with the workflow, visit the logs section by typing "Log" into the search bar on the left hand side of your screen.

Investigating these logs are the best method to troubleshoot what is broken.