# Red Hat Ansible Automation Platform 2.2

## Integrating Automation Services Catalog with your IT Service Management (ITSM) systems

Incorporate Automation Services Catalog into your IT Service Management system toolchain using a workflow defined by order processes and substitutable variables.

Last Updated: 2023-10-31

# Red Hat Ansible Automation Platform 2.2 Integrating Automation Services Catalog with your IT Service Management (ITSM) systems

Incorporate Automation Services Catalog into your IT Service Management system toolchain using a workflow defined by order processes and substitutable variables.

## Abstract

This guide describes the workflows for how to use order processes and substitutable variables to pass data from job templates in you Ansible Automation Platform to an IT Service Management (ITM) system.

# Table of Contents

# PREFACE

You can use the order process features of Automation Services Catalog to integrate with an Information Technology Service Management (ITSM) systems such as ServiceNow.

**IMPORTANT**

Support for automation services catalog is no longer available for Ansible Automation Platform from 2.4.

# MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message .

# CHAPTER 1. PLANNING YOUR ITSM INTEGRATION WITH AUTOMATION SERVICES CATALOG

Plan for the updates and modifications required to your Ansible Automation Platform and playbook inventory to integrate with your ITSM system. Some changes will require specific levels of administrative permissions in both Ansible Tower and Automation Services Catalog.

This section will detail the exact roles required to perform updates, and the artifacts you will need to modify.

**Required permissions**

You will require the following roles to complete the integration workflow:

- You can write or update the playbooks used for ITSM integration.
- You are an Ansible Tower administrator and can create job templates and add surveys.
- You are a Catalog Administrator and can create portfolios, edit surveys, and set order processes.

**Playbooks**

Write or update those playbooks used in the job templates and workflows you intend to use for ITSM integration with Automation Services Catalog.

**Ansible Tower job and workflow template surveys**

Update surveys attached to job templates running the before and after playbooks to support substitutable values.

**Automation Services Catalog**

Update surveys set to products in your ITSM integration workflow to enable substitution.

# CHAPTER 2. UPDATING YOUR PLAYBOOK SET_STATS FIELDS TO SUPPORT INTEGRATION WITH AUTOMATION SERVICES CATALOG

You can update your playbooks to pass information to Automation Services Catalog and support substitutable variables across the product in your order process workflow.

## 2.1. WRITING PLAYBOOK THAT RETURN VALUES TO AUTOMATION SERVICES CATALOG

You can write playbooks designed to return values to Automation Services Catalog. Prefixing **set_stats** parameters with **expose_to_cloud_redhat_com_** will return that value to Automation Services Catalog. You can then pass those values through additional playbooks using substitutable variables.

Use the playbook examples below to learn now about **set_stats** values and to pass values back to Automation Services Catalog and use them as substitutable values in subsequent playbooks in your order processes.

### Example before order playbook

This **before order** playbook returns to Automation Services Catalog **set_stats** values for a favorite color:

```
# This playbook prints a simple debug message and set_stats
- name: Echo Hello, world!
  hosts: localhost
  gather_facts: yes
  tasks:
    - debug:
        msg: "Hello, world!"

    - set_stats:
        data:
          expose_to_cloud_redhat_com_favorite_color: "orange"
```

### Example product order playbook

This playbook passes a substitutable value back to Automation Services Catalog as an artifact value.

```
# This playbook prints a simple debug message with given information
- name: Echo Hello, world!
  hosts: localhost
  gather_facts: yes
  tasks:
    - debug:
        msg: "Hello, {{favorite_color}} world!"

    - set_stats:
        data:
          expose_to_cloud_redhat_com_after_data: "{{favorite_color}}"
```

## Example after order playbook

The after order playbook example below includes the artifact **after_data** value that was passed by the product playbook.

```
# This playbook prints a simple debug message with given information
- name: Echo Hello, world!
  hosts: localhost
  gather_facts: yes
  tasks:
    - debug:
        msg: "{{after_data}}"
~
```

# CHAPTER 3. DEFINING SEQUENTIAL ACTIONS IN ITSM INTEGRATION USING ORDER PROCESSES

Order processes are the primary Automation Services Catalog feature by which you integrate user-ordered products with your ITSM system. Order processes are comprised of products in Automation Services Catalog that are intended to execute job templates that perform actions in your ITSM system. These processes are comprised of **Before order** and **After order** actions. You define your order processes as those products designed to run before and after a product your users order, so that information is passed across the sequence of orders.

**Example sequence**

- **before order** - a product that creates a ticket in the ITSM system.
- **product order** - a web server.
- **after order** - a product that closes the ticket in the ITSM system.

You define the sequence and what products are run by creating an order process that you apply to either your products or at the portfolio level.

For this example workflow, we need to create a order processes and attach them to our **before order**, **product order**, and **after order**.

## 3.1. CREATING AN ORDER PROCESS

As a catalog administrator, create order processes to allow you to execute your Ansible Tower playbooks before and after a product is ordered.

**Prerequisites**

- To create an order process with your Ansible Tower playbooks, you must have an Ansible Tower cluster added as a source.

**Procedure**

1. Select **Order Processes** from the main navigation.
2. Click **Create**.
3. Provide an order process name and description.
4. From the drop-down menus, select actions that will occur before and after an order provision.

   **NOTE**

   The drop-down menus will display playbooks pulled from your Ansible Tower source. Only one Before and After action is supported for each order process.

5. Review the new Order Process details, then click **Create**.

## 3.2. SETTING AN ORDER PROCESS FOR A PRODUCT

Set an order process that will apply to a single product.

**Procedure**

1. Select **Products** from the main navigation, then select a product.
2. Click **More actions** and select **Set order processes**.
3. Expand the drop-down menu and select an order process.

   **NOTE**

   Ansible Automation Platform currently only supports one order process per product provision.

4. Click **Save**.

# CHAPTER 4. ENABLING SUBSTITUTABLE VARIABLES ACROSS THE ORDER PROCESS WORKFLOW

You can substitute variables across products used in your order process workflow. Implementing substitutable variables requires creating or editing the job template survey on Ansible Tower and enabling substitution on surveys attached to products on Automation Services Catalog.

### Substitution format and requirements

Substitutable values are expressed in the format **{{substitution_express}}** with no spaces allowed between the braces and the expression. Refer to Substitute variables for information on each variable you can use for substitution.

## 4.1. IMPLEMENTING SUBSTITUTABLE VARIABLES IN YOUR ORDER PROCESS WORKFLOW

In this section we will demonstrate how to align data you expose to Automation Services Catalog in your playbooks with substitutable variables, using the playbooks examples provided earlier in this guide.

### 4.1.1. Creating a before order process to pass a value to the product order.

**Example playbook before_order.yml for before order process**

```
# This playbook prints a simple debug message and set_stats
- name: Echo Hello, world!
  hosts: localhost
  gather_facts: yes
  tasks:
    - debug:
        msg: "Hello, world!"

    - set_stats:
        data:
          expose_to_cloud_redhat_com_favorite_color: "orange"
```

This playbook returns to Automation Services Catalog a playbook artifact **favorite_color** that has the value **orange**. You do not need to make a job template survey on Ansible Tower since no user input is required. Instead, you can pass **favorite_color** to the product playbook by editing the survey on Automation Services Catalog set to the before order process product.

### 4.1.2. Configuring the product order to use a substituted variable

This example product playbook accepts a substituted variable for {{favorite_color}} and passes back to Automation Services Catalog the artifact 'after_data'.

**Example playbook product_order.yml**

```
# This playbook prints a simple debug message with given information
- name: Echo Hello, world!
  hosts: localhost
  gather_facts: yes
  tasks:
    - debug:
        msg: "Hello, {{favorite_color}} world!"

    - set_stats:
        data:
          expose_to_cloud_redhat_com_after_data: "{{favorite_color}}"
```

Create a job template on Ansible Tower for the 'product_order.yml' playbook that sets **favorite_color** as the *Answer Variable Name.

Create a survey for the **product_order.yml** job template that includes:

1. Enter **What is your favorite color?** in the **Prompt** field.
2. Enter **favorite_color** in the **Answer Variable Name** field.
3. Click **UPDATE**.

Enable the **product order** survey on Automation Services Catalog to use the 'favorite_color' value passed from **before_order.yml**.

In the survey attached to your **product order**:

1. Update fields with the following values:
   a. **Name**: favorite_color
   b. **Initial Value**: {{before.before_order.artifacts.favorite_color}}
   c. **Label**: What is your favorite color?
   d. **Placeholder**: {{before.before_order.artifacts.favorite_color}}
2. Toggle the **Disabled** switch to the active position.
3. Toggle the **Substitution** switch to the active position.

You have now configured the **product order** to use the **favorite_color** artifact from the **before_order.yml** playbook as a substituted variable.

### 4.1.3. Configuring the after order product to accept after_data artifact values from the product order

Use surveys to pass the 'product_order.yml' artifact **after_data** to **after_order.yml** using substitutable variables.

**Example playbook after_order.yml for after order product**

```
# This playbook prints a simple debug message with given information
- name: Echo Hello, world!
  hosts: localhost
  gather_facts: yes
  tasks:
    - debug:
        msg: "{{after_data}}"
```

-

Create a job template on Ansible Tower for the 'after_order.yml' playbook that sets **after_data** as the *Answer Variable Name.

In the survey attached to your **after_order.yml** job template:

1. Type "Enter your after data" in the **Prompt** field.
2. Enter **after_data** in the **Answer Variable Name** field.
3. Click **UPDATE**.

Enable the **after order** survey on Automation Services Catalog to use the 'after_data' value passed from **after_order.yml**.

In the survey attached to your **after order**:

1. Update fields with the following values:
    a. **Name**: after_data
    b. **Initial Value**: {{product.artifacts.after_data}}
    c. **Label**: "Enter your after data"
    d. **Placeholder**: {{product.artifacts.after_data}}
2. Toggle the **Disabled** switch to the active position.
3. Toggle the **Substitution** switch to the active position.

The playbook will display the value passed for **after data** when the job template is run.

## 4.2. SUBSTITUTE VARIABLES REFERENCE

A substitution has the format of **{{substitution_express}}**. Do not use a space between the curly brackets and the expression.

Substituted values are set in Ansible Tower jobs template surveys. Substituted variables pass from one product in the order process to another, depending on your configuration.

See Surveys in the Ansible Tower *User Guide* for more information on updating job template surveys.

Follow the table below when including substitutable variables in your surveys.

**Table 4.1. Object model reference table**

| Object | Expression | Notes |
|---|---|---|

| Object | Expression | Notes |
|---|---|---|
| Order | **order.order_id**<br><br>**order.created_at**<br><br>**order.ordered_by.name**<br><br>**order.ordered_by.email**<br><br>**order.approval.updated_at**<br><br>**order.approval.decision**<br><br>**order.approval.reason** | order.created_at` is the date and time when the order started.<br><br>Date format: **2007-02-10T20:30:45Z**<br><br>Example: {{order.approval.reason}} |
| Before order | **before.<order_process_name>.artifacts.<fact_name>**<br><br>**before.<order_process_name>.parameters.<parameter_name>**<br><br>**before.<order_process_name>.status** | No quotes are needed for the order_process_name if it contains space.<br><br>Example: {{before.Sample Order.artifacts.ticket_number}} |
| Product | **product.name**<br><br>**product.description**<br><br>**product.long_description**<br><br>**product.help_url**<br><br>**product.support_url**<br><br>**product.vendor**<br><br>**product.platform**<br><br>**product.portfolio.name**<br><br>**product.portfolio.description**<br><br>**product.artifacts.<fact_name>**<br><br>**product.parameters.<parameter_name>**<br><br>**product.status** | Example: {{product.artifacts.after_data}} |

| Object | Expression | Notes |
|---|---|---|
| After order | **after.<order_process_name>.artifacts.<fact_name>**<br><br>**after.<order_process_name>.parameters.<parameter_name>**<br><br>**after.<order_process_name>.status** | No quotes are needed for the order_process_name if it contains space.<br><br>Example: {{after.SampleOrder.status}} |