



Red Hat Ansible Automation Platform 2.5

Getting started with Ansible Automation Platform

Get started with Ansible Automation Platform

Red Hat Ansible Automation Platform 2.5 Getting started with Ansible Automation Platform

Get started with Ansible Automation Platform

Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This guide shows how to get started with Ansible Automation Platform.

Table of Contents

PREFACE	4
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	5
CHAPTER 1. KEY FUNCTIONALITY AND CONCEPTS	6
1.1. ACTIVITY STREAM	6
1.2. AUTOMATION EXECUTION	6
1.2.1. Inventories	6
1.3. AUTOMATION CONTENT	6
1.3.1. Ansible roles	7
1.3.2. Ansible playbooks	7
1.4. AUTOMATION DECISIONS	8
1.5. AUTOMATION MESH	8
1.6. RED HAT ANSIBLE LIGHTSPEED	8
1.7. ANSIBLE DEVELOPMENT TOOLS	8
1.8. ANSIBLE AUTOMATION PLATFORM INSTALLATION AND CONFIGURATION	9
1.9. DASHBOARD COMPONENTS	9
1.10. USING THIS GUIDE	10
CHAPTER 2. GETTING STARTED AS A PLATFORM ADMINISTRATOR	11
2.1. LOGGING IN FOR THE FIRST TIME	11
2.2. CONFIGURE AUTHENTICATION	12
2.3. MANAGING USER ACCESS WITH ROLE-BASED ACCESS CONTROL	12
2.4. CREATING AN ORGANIZATION	12
2.5. CREATING A TEAM	13
2.6. CREATING A USER	13
2.7. CONFIGURING GITHUB AUTHENTICATION	14
CHAPTER 3. GETTING STARTED AS AN AUTOMATION DEVELOPER	16
3.1. SETTING UP YOUR DEVELOPMENT ENVIRONMENT	16
3.2. CREATE AUTOMATION CONTENT WITH PLAYBOOKS	16
3.2.1. Create a playbook	16
3.3. DEFINE EVENTS WITH RULEBOOKS	17
3.3.1. Rulebook actions	17
3.4. BUNDLE CONTENT WITH ANSIBLE ROLES	17
3.4.1. Creating a role	18
3.5. CREATING A ROLE	18
3.6. ABOUT CONTENT COLLECTIONS	19
3.6.1. Downloading content	19
3.7. PUBLISHING TO A COLLECTION	19
3.7.1. Uploading a collection to automation hub	20
3.8. BUILD AND USE AN EXECUTION ENVIRONMENT	21
3.8.1. Using the base automation execution environment	21
3.8.1.1. Customize the base execution environment image	22
3.8.1.2. About Ansible Builder	22
3.8.2. Adding an execution environment to a job template	22
3.8.2.1. About container registries	24
3.9. BUILD AND USE A DECISION ENVIRONMENT	24
3.9.1. Setting up a new decision environment	24
3.10. CREATING AN AUTOMATION EXECUTION PROJECT	25
3.11. CREATING AN AUTOMATION DECISION PROJECT	26
3.12. ABOUT INVENTORIES	27

3.12.1. Browsing and creating inventories	27
3.12.2. Browsing and creating inventories	28
3.13. WORK WITH JOB TEMPLATES	30
3.13.1. Getting started with job templates	30
3.13.2. Creating a job template	30
3.13.3. Running a job template	38
3.13.4. Editing a job template	38
3.14. CREATE AND RUN A RULEBOOK ACTIVATION	39
3.14.1. Setting up a rulebook activation	39
3.14.1.1. Rulebook activation list view	41
3.14.2. Enabling and disabling rulebook activations	42
3.14.3. Restarting rulebook activations	42
3.14.4. Deleting rulebook activations	42
3.14.5. Activating webhook rulebooks	43
CHAPTER 4. GETTING STARTED AS AN AUTOMATION OPERATOR	45
4.1. GET STARTED WITH PLAYBOOKS	45
4.1.1. Learn about playbooks	45
4.2. WRITING A PLAYBOOK	45
4.3. BUNDLE CONTENT WITH ANSIBLE ROLES	46
4.3.1. Creating a role	47
4.3.2. Creating a role	47
4.4. ABOUT AUTOMATION CONTENT	48
4.4.1. About content collections	48
4.4.1.1. Downloading content	48
4.4.2. Browse content	48
4.4.3. Downloading content	49
4.5. PUBLISHING TO A COLLECTION	49
4.5.1. Manage collections in automation hub	50
4.5.2. Uploading a collection to automation hub	50
4.6. BUILD AND USE AN EXECUTION ENVIRONMENT	51
4.6.1. Using the base automation execution environment	52
4.6.1.1. Customize the base execution environment image	52
4.6.1.2. About Ansible Builder	52
4.6.2. Adding an execution environment to a job template	53
4.7. AUTOMATION EXECUTION PROJECTS	54
4.7.1. Viewing project details	54
4.8. WORK WITH JOB TEMPLATES	54
4.8.1. Launching a job template	55
4.9. ABOUT INVENTORIES	55
4.9.1. Executing an inventory	55
4.10. AUTOMATION EXECUTION JOBS	56
4.10.1. Reviewing a job status	56
4.10.2. Reviewing job output	56

PREFACE

Red Hat Ansible Automation Platform is a unified automation solution that automates a variety of IT processes, including provisioning, configuration management, application deployment, orchestration, and security and compliance changes (including patching systems).

Ansible Automation Platform features a platform interface where you can set up centralized authentication, configure access management, and execute automation tasks from a single location.

This guide will help you get started with Ansible Automation Platform by introducing three central concepts: automation execution, automation decisions, and automation content.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

If you have a suggestion to improve this documentation, or find an error, you can contact technical support at <https://access.redhat.com> to open a request.

CHAPTER 1. KEY FUNCTIONALITY AND CONCEPTS

With Ansible Automation Platform, you can create, manage, and scale automation for your organization across users, teams, and regions. See the following functionality and concepts of Ansible Automation Platform for more details.

The release of Ansible Automation Platform 2.5 introduces an updated, unified user interface (UI) that allows you to interact with and manage each part of the platform.

Platform gateway is the service that handles authentication and authorization for the Ansible Automation Platform. It provides a single entry into the Ansible Automation Platform and serves the platform user interface so you can authenticate and access all of the Ansible Automation Platform services from a single location. For more information about the services available in the Ansible Automation Platform, refer to [Key functionality and concepts](#) in *Getting started with Ansible Automation Platform*.

1.1. ACTIVITY STREAM

The platform gateway includes an activity stream that captures changes to gateway resources, such as the creation or modification of organizations, users, and service clusters, among others. For each change, the activity stream collects information about the time of the change, the user that initiated the change, the action performed, and the actual changes made to the object, when possible. The information gathered varies depending on the type of change.

You can access the details captured by the activity stream from the API:

```
/api/gateway/v1/activitystream/
```

1.2. AUTOMATION EXECUTION

The centerpiece of Ansible Automation Platform is its automation execution command and control center, where you can deploy, define, operate, scale and delegate automation across your enterprise. With this functionality, you can perform a variety of tasks from a single location, such as running playbooks from a simple, straightforward web UI, monitoring dashboard activity, and centralized logging to manage and track job execution.

In the automation execution environment, you can use automation controller tasks to build job templates, which standardize how automation is deployed, initiated, and delegated, making it more reusable and consistent.

1.2.1. Inventories

An inventory is a single file, usually in INI or YAML format, containing a list of hosts and groups that can be acted upon using Ansible commands and playbooks. You can use an inventory file to specify your installation scenario and describe host deployments to Ansible. You can also use an inventory file to organize managed nodes in centralized files that give Ansible with system information and network locations.

1.3. AUTOMATION CONTENT

Automation hub is the central location for your Ansible Automation Platform content. In automation hub you can also find content collections that you can download and integrate into your automation environment. You can also create and upload your own content to distribute to your users.

An Ansible Content Collection is a ready-to-use toolkit for automation and can include multiple types of content, including roles, modules, playbooks, and plugins all in one place.

You can access automation hub in one of two ways:

- On the Red Hat-hosted [Hybrid Cloud Console](#), where you can find Red Hat validated or certified content that you can sync to your platform environment.
- On a self-hosted, on-premise private automation hub, where you can curate content for your automation users and manage access to collections and execution environments.

Depending on the way you access automation hub, you may have access to different types of content collections.

There are two types of Red Hat Ansible content:

- Ansible Certified Content Collections, which Red Hat builds, supports, and maintains. Certified collections are included in your subscription to Red Hat Ansible Automation Platform and can be found in automation hub.
- Ansible validated content collections, which are customizable and therefore do not have a support guarantee, but have been tested in the Ansible Automation Platform environment.

For more information about Ansible content, see [Create automation content](#) in [Getting started as an automation developer](#).

1.3.1. Ansible roles

Ansible roles allow you to create reusable automation content that helps teams to work more efficiently and avoid duplicating efforts. With roles, you can group together a broader range of existing automation content, like playbooks, configuration files, templates, tasks, and handlers to create customized automation content that can be reused and shared with others.

You can also make roles configurable by exposing variables that users can set when calling the role, allowing them to configure their system according to their organization's requirements.

Roles are generally included in Ansible content collections.

Additional resources

For more information, see [Bundle content with Ansible roles](#).

1.3.2. Ansible playbooks

Playbooks are YAML files that contain specific sets of human-readable instructions, or "plays," that you send to run on a single target or groups of targets. Ansible playbooks are repeatable and reusable configuration management tools designed to deploy complex applications.

You can use playbooks to manage configurations of and deployments to remote machines to sequence multitiered rollouts involving rolling updates. Use playbooks to delegate actions to other hosts, interacting with monitoring servers and load balancers along the way.

Once written, you can use and re-use playbooks for automation across your enterprise. For example, if you need to run a task more than once, write a playbook and put it under source control. Then, you can use the playbook to push out new configuration or confirm the configuration of remote systems.

Ansible playbooks can declare configurations, orchestrate steps of any manually ordered process on many machines in a defined order, or start tasks synchronously or asynchronously.

You can also use Red Hat Ansible Lightspeed, Ansible's generative AI service, to create and develop playbooks to fit your needs. See the [Ansible Lightspeed documentation](#) for more information.

Additional resources

- [Getting started with playbooks](#)
- [Red Hat Ansible Lightspeed with IBM watsonx Code Assistant user guide](#)

1.4. AUTOMATION DECISIONS

Red Hat Ansible Automation Platform includes Event-Driven Ansible, an automation engine that listens to your system's event stream and reacts to events that you have specified with targeted automation tasks. In this way, Event-Driven Ansible manages routine automation tasks and responses, freeing you up to work on more complex tasks.

Managed through Event-Driven Ansible controller, Ansible rulebooks are the framework for automation decisions. Ansible rulebooks are collections of rulesets, which in turn consist of one or more sources, rules, and conditions. Rulebooks tell the system what events to flag and how to respond to them. From the Automation Decisions section of the platform user interface, you can use rulebooks to connect and listen to event sources, and define actions that are triggered in response to certain events.

Additional resources

For more information about rulebook, events, and sources, see [Rulebook actions](#).

1.5. AUTOMATION MESH

Automation mesh is an overlay network intended to ease the distribution of automation across a collection of execution nodes using existing connectivity. Execution nodes are where [Ansible Playbooks](#) are actually executed. A node runs an automation execution environment which, in turn, runs the Ansible Playbook. Automation mesh creates peer-to-peer connections between these execution nodes, increasing the resiliency of your automation workloads to network latency and connection disruptions. This also permits more flexible architectures and provides rapid, independent scaling of control and execution capacity.

1.6. RED HAT ANSIBLE LIGHTSPEED

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant is a generative AI service designed by and for Ansible platform engineers and developers. It accepts natural-language prompts entered by a user and then interacts with IBM watsonx foundation models to produce code recommendations built on Ansible best practices.

Red Hat Ansible Lightspeed with IBM watsonx Code Assistant helps automation teams learn, create, and maintain Red Hat Ansible Automation Platform content more efficiently.

1.7. ANSIBLE DEVELOPMENT TOOLS

Ansible development tools are an integrated and supported suite of capabilities that help IT practitioners at any skill level generate automation content faster than they might with manual coding. Ansible development tools can help you create, test, and deploy automation content like playbooks,

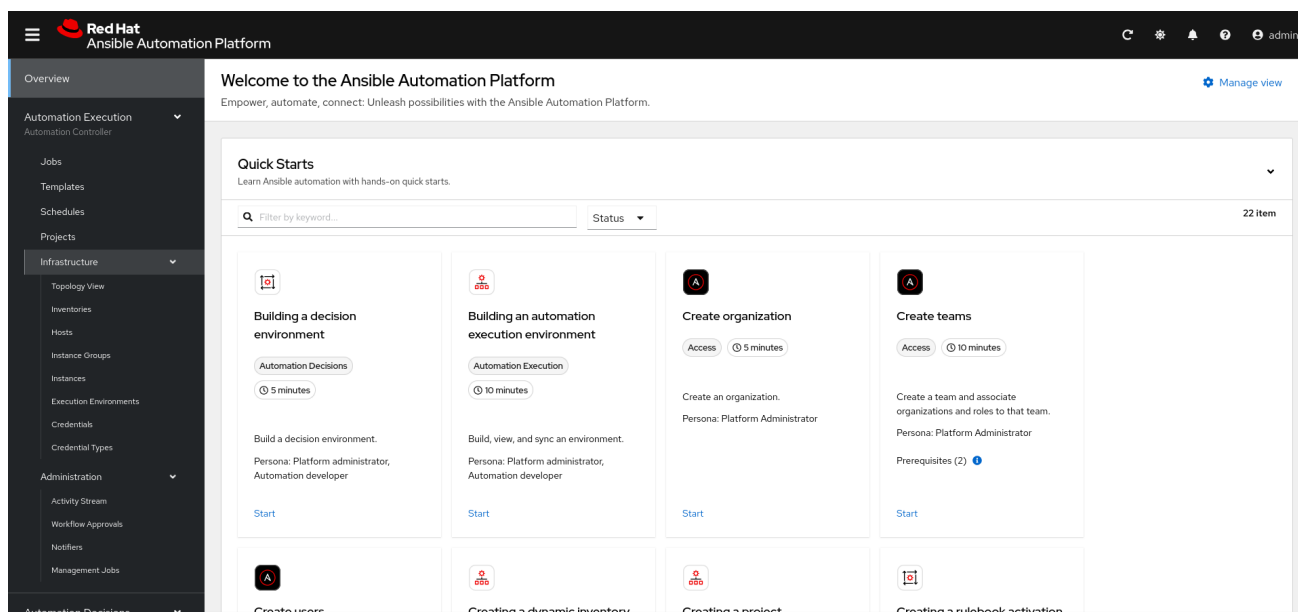
execution environments, and collections quickly and accurately using recommended practices. For more information on how Ansible development tools can help you create automation content, see our documentation on [Developing automation content](#).

1.8. ANSIBLE AUTOMATION PLATFORM INSTALLATION AND CONFIGURATION

Red Hat Ansible Automation Platform offers flexible installation and configuration options. Depending on your organization's needs, you can install Red Hat Ansible Automation Platform using one of the following methods, based on your environment:

- [RPM installation](#)
- [Installing on OpenShift Container Platform](#)
- [Cloud environments](#)
- [Containerized installation](#)

1.9. DASHBOARD COMPONENTS



After you install Ansible Automation Platform on your system and log in for the first time, familiarize yourself with the platform dashboard.

Quick starts

You can learn about Ansible automation functions with guided tutorials called quick starts. In the dashboard, you can access quick starts by selecting a quick start card. From the panel displayed, click **Start** and complete the onscreen instructions. You can also filter quick starts by keyword and status.

Resource status

Indicates the status of your hosts, projects, and inventories. The status indicator links to your configured hosts, projects and inventories where you can search, filter, add and change these resources.

Job Activity

You can view a summary of your current job status. Filter the job status within a period of time or by job type, or click **Go to jobs** to view a complete list of jobs that are currently available.

Jobs

You can view recent jobs that have run, or click **View all Jobs** to view a complete list of jobs that are currently available, or create a new job.

Projects

You can view recently updated projects or click **View all Projects** to view a complete list of the projects that are currently available, or create a new project.

Inventories

You can view recently updated inventories or click **View all Inventories** to view a complete list of available inventories, or create a new inventory.

Rulebook Activations

You can view the list of recent rulebook activations and their status, display the complete list of rulebook activations that are currently available, or create a new rulebook activation.

Rule Audit

You view recently fired rule audits, view rule audit records, and view rule audit data based on corresponding rulebook activation runs.

Decision Environments

You can view recently updated decision environments, or click **View all Decision Environments** to view a complete list of available inventories, or create a new decision environment.

1.10. USING THIS GUIDE

After you have installed Ansible Automation Platform 2.5 and have become familiar with the dashboard, use this document to explore further options for setup and daily use. This guide is structured so that you can select the path that is most appropriate to you and your role within your organization. We also encourage you to explore the other paths outlined in this guide to learn how Ansible empowers users with various roles and objectives to build and customize automation tasks.

Select one of the following paths to continue getting started:

- If you are a systems administrator configuring authentication and setting up teams and organizations, see [Getting started as a platform administrator](#) .
- If you are a developer setting up development environments, creating playbooks, rulebooks, roles, or projects, see [Getting started as an automation developer](#) .
- If you are an operator using playbooks, publishing custom content, creating projects, and creating and using inventories, see [Getting started as an automation operator](#) .

CHAPTER 2. GETTING STARTED AS A PLATFORM ADMINISTRATOR

As a platform administrator, Ansible Automation Platform can help you enable your users and teams to develop and run automation.

This guide walks you through the basic steps to get set up as an administrator for Ansible Automation Platform, including configuring and maintaining the platform for users.

To get started as an administrator, see the following:

- [Logging in for the first time](#)
- [Configure authentication](#)
- [Managing user access with role-based access control](#)

2.1. LOGGING IN FOR THE FIRST TIME

Log in to the Ansible Automation Platform as an administrator and enter your subscription information. You can then create user profiles and assign roles.

Procedure

1. With the login information provided after your installation completed, open a web browser and log in to Red Hat Ansible Automation Platform by navigating to its server URL at:
https://<AAP_SERVER_NAME>
2. Use the credentials specified during the installation process to login:
 - The default username is **admin**.
 - The password for **admin** is the value specified during installation.

After your first login, you are prompted to add your subscription manifest.

Procedure

1. You can select between uploading a copy of your subscription manifest or entering your login credentials to find the subscription associated with your profile:
 - a. To upload a subscription manifest, drag the file to the field beneath **Red Hat subscription manifest** or browse for the file on your local machine.
 - b. To find your subscription, click the tab labeled **Username / password** and enter your credentials. Your subscription appears in the list menu labeled **Subscription**. Select your subscription.
2. After you have added your subscription, click **Next**.
3. On the screen labeled **Analytics**, check the box if you want to share data with Red Hat and click **Next**.
4. Check the box indicating that you agree to the **End User License Agreement**
5. Review your information and click **Finish**.

TIP

After logging in, review the quick starts section for useful guidance.

2.2. CONFIGURE AUTHENTICATION

After your first login as an administrator you must configure authentication for your users. Depending on your organization's needs and resources, you can either:

- Set up authentication by creating users, teams, and organizations manually.
- Use an external source such as GitHub to configure authentication for your system.

2.3. MANAGING USER ACCESS WITH ROLE-BASED ACCESS CONTROL

Role-based access control (RBAC) restricts user access based on their role within an organization. The roles in RBAC refer to the levels of access that users have to the network.

You can control what users can do with the components of Ansible Automation Platform at a broad or granular level depending on your RBAC policy. You can select whether the user is a system administrator or normal user and align roles and access permissions with their positions within the organization.

You can define roles with many permissions that can then be assigned to resources, teams, and users. The permissions that make up a role dictate what the assigned role allows. Permissions are allocated with only the access needed for a user to perform the tasks appropriate for their role.

2.4. CREATING AN ORGANIZATION

Ansible Automation Platform automatically creates a default organization. If you have a self-support level license, you have only the default organization available and cannot delete it.

Procedure

1. From the navigation panel, select **Access Management** → **Organizations**.
2. Click **Create organization**.
3. Enter the **Name** and optionally provide a **Description** for your organization.



NOTE

If automation controller is enabled on the platform, continue with Step 4. Otherwise, proceed to Step 6.

4. Select the name of the **Execution environment** or search for one that exists that members of this team can run automation.
5. Enter the name of the **Instance Groups** on which to run this organization.
6. Optional: Enter the **Galaxy credentials** or search from a list of existing ones.

7. Select the **Max hosts** for this organization. The default is 0. When this value is 0, it signifies no limit. If you try to add a host to an organization that has reached or exceeded its cap on hosts, an error message displays:

You have already reached the maximum number of 1 hosts allowed for your organization. Contact your System Administrator for assistance.

8. Click **Next**.
9. If you selected more than 1 instance group, you can manage the order by dragging and dropping the instance group up or down in the list and clicking **Confirm**.



NOTE

The execution precedence is determined by the order in which the instance groups are listed.

10. Click **Next** and verify the organization settings.
11. Click **Finish**.

2.5. CREATING A TEAM

You can create new teams, assign an organization to the team, and manage the users and administrators associated with each team. Users associated with a team inherit the permissions associated with the team and any organization permissions to which the team has membership.

To add a user or administrator to a team, the user must have already been created.

Procedure

1. From the navigation panel, select **Access Management** → **Teams**.
2. Click **Create team**.
3. Enter a **Name** and optionally give a **Description** for the team.
4. Select an **Organization** to be associated with this team.



NOTE

Each team can only be assigned to one organization.

5. Click **Create team**.
The **Details** page opens, where you can review and edit your team information.

2.6. CREATING A USER

There are three types of users in Ansible Automation Platform:

Normal user

Normal users have read and write access limited to the resources (such as inventory, projects, and job templates) for which that user has been granted the appropriate roles and privileges. Normal users are the default type of user.

Ansible Automation Platform Administrator

An administrator (also known as a Superuser) has full system administration privileges – with full read and write privileges over the entire installation. An administrator is typically responsible for managing all aspects of and delegating responsibilities for day-to-day work to various users.

Ansible Automation Platform Auditor

Auditors have read-only capability for all objects within the environment.

Procedure

1. From the navigation panel, select **Access Management** → **Users**.
2. Click **Create user**.
3. Enter the details about your new user in the fields on the **Create** user page. Fields marked with an asterisk (*) are required.
4. Normal users are the default when no User type is specified. To define a user as an administrator or auditor, select a **User type** checkbox.

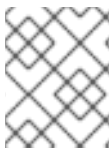


NOTE

If you are modifying your own password, log out and log back in again for it to take effect.

5. Select the **Organization** to be assigned for this user. For information about creating a new organization, refer to [Creating an organization](#).
6. Click **Create user**.

When the user is successfully created, the **User** dialog opens. From here, you can review and modify the user's Teams, Roles, Tokens and other membership details.



NOTE

If the user is not newly-created, the details screen displays the last login activity of that user.

If you log in as yourself, and view the details of your user profile, you can manage tokens from your user profile by selecting the **Tokens** tab.

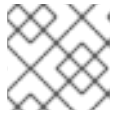
2.7. CONFIGURING GITHUB AUTHENTICATION

You can connect GitHub identities to Ansible Automation Platform using OAuth. To set up GitHub authentication, you need to obtain an OAuth2 key and secret by registering your organization-owned application from GitHub using the [registering the new application with GitHub](#).

The OAuth2 key (Client ID) and secret (Client Secret) are used to supply the required fields in the UI. To register the application, you must supply it with your webpage URL, which is the Callback URL shown in the Authenticator details for your authenticator configuration.

Procedure

1. From the navigation panel, select **Access Management** → **Authentication Methods**.
2. Click **Create authentication**.
3. Select **GitHub** from the **Authentication type** list and click **Next**.
4. Enter a **Name** for this authentication configuration.
5. When the application is registered, GitHub displays the **Client ID** and **Client Secret**:
 - a. Copy and paste the GitHub Client ID into the GitHub OAuth2 Key field.
 - b. Copy and paste the GitHub Client Secret into the GitHub OAuth2 Secret field.
6. Optional: Enter any **Additional Authenticator Fields** that this authenticator can take. These fields are not validated and are passed directly back to the authenticator.



NOTE

Values defined in this field override the dedicated fields provided in the UI.

7. To automatically create organizations, users, and teams upon successful login, select **Create objects**.
8. To enable this authentication method upon creation, select **Enabled**.
9. To remove a user for any groups they were previously added to when they authenticate from this source, select **Remove users**.
10. Click **Next**.

Verification

To verify that the authentication is configured correctly, log out of Ansible Automation Platform and check that the login screen displays the logo of your authentication chosen method to enable logging in with those credentials.

Next steps

To control which users are allowed into the Ansible Automation Platform server, and placed into Ansible Automation Platform organizations or teams based on their attributes (like username and email address) or to what groups they belong, continue to [Mapping](#).

CHAPTER 3. GETTING STARTED AS AN AUTOMATION DEVELOPER

As an automation developer, you can use Ansible Automation Platform to implement your organization's automation strategy. Ansible Automation Platform can help you write, test, and share automation content, and download and use Red Hat certified collections. This guide will walk you through the basic steps to get set up as an automation developer on Ansible Automation Platform, with information on how to:

- Set up your development environment
- Create, publish, and use custom automation content
- Build and use automation execution environments and decision environments
- Create and run rulebook activations for Event-Driven Ansible
- Create and use job templates

3.1. SETTING UP YOUR DEVELOPMENT ENVIRONMENT

Before you begin to create content, consult our guide to [Developing automation content](#). There you can find information on Ansible development tools, which you can integrate into your environment, and learn how to scaffold a playbook project.

3.2. CREATE AUTOMATION CONTENT WITH PLAYBOOKS

Ansible playbooks are blueprints that tell Ansible Automation Platform what tasks to perform with which devices. You can use a playbook to define the automation tasks that you want the platform to run.

3.2.1. Create a playbook

A playbook contains one or more plays. A basic play contains the following parameters:

- **Name:** a brief description of the overall function of the playbook, which assists in keeping it readable and organized for all users.
- **Hosts:** identifies the target or targets for Ansible to run against.
- **Become statements:** this optional statement can be set to **true** or **yes** to enable privilege escalation using a become plugin (such as **sudo**, **su**, **pfexec**, **doas**, **pbrun**, **dzdo**, **ksu**).
- **Tasks:** this is the list of actions that get executed against each host in the play.

Here is an example of a play in a playbook. You can see the name of the play, the host, and the list of tasks included in the play.

```
- name: Set Up a Project and Job Template
hosts: host.name.ip
become: true

tasks:
- name: Create a Project
  ansible.controller.project:
```

```

name: Job Template Test Project
state: present
scm_type: git
scm_url: https://github.com/ansible/ansible-tower-samples.git

```

```

- name: Create a Job Template
  ansible.controller.job_template:
    name: my-job-1
    project: Job Template Test Project
    inventory: Demo Inventory
    playbook: hello_world.yml
    job_type: run
    state: present

```

For more detailed instructions on authoring playbooks, see [Developing automation content](#), or consult our documentation on [Red Hat Ansible Lightspeed with IBM watsonx Code Assistant User Guide](#) to learn how to generate a playbook with AI assistance.

3.3. DEFINE EVENTS WITH RULEBOOKS

An Ansible rulebook is a collection of rulesets that references one or more sources, rules, and conditions.

Rulebooks are to Event-Driven Ansible what playbooks are to Ansible Automation Platform as a whole. Like a playbook, a rulebook defines automation tasks for the platform, along with when they should be triggered.

3.3.1. Rulebook actions

Rulebooks use an "if-this-then-that" logic that tells Event-Driven Ansible what actions to activate when a rule is triggered. Event-Driven Ansible listens to the controller event stream and, when an event triggers a rule, activates an automation action in response.

Rulebooks can trigger the following activations:

- **run_job_template**
- **run_playbook** (only supported with ansible-rulebook CLI)
- **debug**
- **print_event**
- **set_fact**
- **post_event**
- **retract_fact**
- **shutdown**

To read more about rulebook activations, see [Actions](#) in the Ansible Rulebook documentation.

3.4. BUNDLE CONTENT WITH ANSIBLE ROLES

A role is like a customized piece of automation content that bundles together relevant bits from

playbooks to fit your system's specific needs. Roles are self-contained and portable, and can include groupings of tasks, variables, configuration templates, handlers, and other supporting files to orchestrate complicated automation flows.

Instead of creating huge playbooks with hundreds of tasks, you can use roles to break the tasks apart into smaller, more discrete units of work.

To learn more about roles, see [What is an Ansible Role—and how is it used?](#) .

3.4.1. Creating a role

You can create roles using the Ansible Galaxy CLI tool, which is included with your Ansible Automation Platform bundle. Access role-specific commands from the roles subcommand:

```
ansible-galaxy role init <role_name>
```

Standalone roles outside of collections are still supported, but we recommend that you create new roles inside of a collection to take advantage of all the features Ansible Automation Platform has to offer.

3.5. CREATING A ROLE

Procedure

1. In your terminal, navigate to the roles directory inside a collection.
2. Create a role called **my_role** inside the collection:

```
$ansible-galaxy role init my_role
```

The collection now includes a role named **my_role** inside the **roles** directory, as you can see in this example:

```
~/.ansible/collections/ansible_collections/<my_namespace>/<my_collection_name>
...
├── roles/
│   └── my_role/
│       ├── .travis.yml
│       ├── README.md
│       ├── defaults/
│       │   └── main.yml
│       ├── files/
│       ├── handlers/
│       │   └── main.yml
│       ├── meta/
│       │   └── main.yml
│       ├── tasks/
│       │   └── main.yml
│       ├── templates/
│       ├── tests/
│       │   ├── inventory
│       │   └── test.yml
│       └── vars/
│           └── main.yml
```

3. A custom role skeleton directory can be supplied by using the **--role-skeleton** argument. This allows organizations to create standardized templates for new roles to suit their needs.

```
$ansible-galaxy role init my_role --role-skeleton ~/role_skeleton
```

This creates a role named **my_role** by copying the contents of `~/role_skeleton` into **my_role**. The contents of **role_skeleton** can be any files or folders that are valid inside a role directory.

3.6. ABOUT CONTENT COLLECTIONS

Ansible content collections are assemblages of automation content. There are two types of Ansible collections:

- **Ansible Certified Content Collections**, which contain fully-supported roles and modules that are enterprise- and production-ready for use in your environments.
- **Ansible validated content collections** which provide you with a trusted, expert-guided approach for performing foundational operations and tasks in your product.

Both types of content collections can be found in automation hub through the [Hybrid Cloud Console](#).

3.6.1. Downloading content

After collections are finalized, you can import them to a location where they can be distributed to others across your organization.

Procedure

1. Log in to Red Hat Ansible Automation Platform.
2. From the navigation panel, select **Automation Content** → **Collections**. The **Collections** page displays all collections across all repositories. You can search for a specific collection.
3. Select the collection that you want to export. The collection details page opens.
4. From the **Install** tab, select **Download tarball**. The .tar file is downloaded to your default browser downloads folder. You can now import it to the location of your choosing.

3.7. PUBLISHING TO A COLLECTION

You can configure your projects to be uploaded to Git, or to the source control manager of your choice.

Procedure

1. From the navigation panel, select **Automation Execution** → **Projects**.
2. Locate or create the project that you want to publish to your source control manager.
3. In the project **Details** tab, select **Edit project**.
4. Select **Git** from the **Source Control Type** drop-down menu.
5. Enter the appropriate details into the following fields:
 - a. **Source Control URL** - see an example in the tooltip.

- b. Optional: **Source control branch/tag/commit**: Enter the SCM branch, tags, commit hashes, arbitrary refs, or revision number (if applicable) from the source control to checkout. Some commit hashes and references might not be available unless you also provide a custom refs spec in the next field. If left blank, the default is **HEAD**, which is the last checked out branch, tag, or commit for this project.
 - c. **Source Control Refspec** - This field is an option specific to Git source control and only advanced users familiar and comfortable with Git should specify which references to download from the remote repository. For more information, see [Job branch overriding](#).
 - d. **Source Control Credential** - If authentication is required, select the appropriate source control credential.
6. Optional: **Options** - select the launch behavior, if applicable:
- a. **Clean** - Removes any local modifications before performing an update.
 - b. **Delete** - Deletes the local repository in its entirety before performing an update. Depending on the size of the repository this can significantly increase the amount of time required to complete an update.
 - c. **Track submodules** - Tracks the latest commit. See the tooltip for more information.
 - d. **Update Revision on Launch** - Updates the revision of the project to the current revision in the remote source control, and caches the roles directory from [Ansible Galaxy](#) or [Collections support](#). Automation controller ensures that the local revision matches and that the roles and collections are up-to-date with the last update. In addition, to avoid job overflows if jobs are spawned faster than the project can synchronize, selecting this enables you to configure a cache timeout to cache previous project synchronizations for a given number of seconds.
 - e. **Allow Branch Override** - Enables a job template or an inventory source that uses this project to start with a specified SCM branch or revision other than that of the project. For more information, see [Job branch overriding](#).
7. Click **Save** to save your project.

3.7.1. Uploading a collection to automation hub

If you want to share a collection that you have created with the rest of the Ansible community, you can upload it to automation hub.



NOTE

Sharing a collection with the Ansible community requires getting the collection certified or validated by our Partner Engineering team. This action is available only to partner clients. For more about becoming a partner, see our [documentation on software certification](#).

You can upload your collection by using either the automation hub user interface or the **ansible-galaxy** client.

Prerequisites

- You have configured the **ansible-galaxy** client for automation hub.

- You have at least one namespace.
- You have run all content through **ansible-test sanity**

Procedure

1. From the navigation panel, select **Automation Content** → **Namespaces**.
2. Within the My namespaces tab, locate and click into the namespace to which you want to upload a collection.
3. Select the **Collections** tab, and then click **Upload collection**.
4. In the New collection modal, click **Select file**. Locate the file on your system.
5. Click **Upload**.

Using the **ansible-galaxy** client, enter the following command:

```
$ ansible-galaxy collection publish path/to/my_namespace-my_collection-1.0.0.tar.gz --api-key=SECRET
```

3.8. BUILD AND USE AN EXECUTION ENVIRONMENT

All automation in Red Hat Ansible Automation Platform runs on container images called automation execution environments.

Automation execution environments are consistent and shareable container images that serve as Ansible control nodes. Automation execution environments reduce the challenge of sharing Ansible content that has external dependencies. If automation content is like a script that a developer has written, an automation execution environment is like a replica of that developer's environment, thereby enabling you to reproduce and scale the automation content that the developer has written. In this way, execution environments make it easier for you to implement automation in a range of environments.

Automation execution environments contain:

- Ansible Core
- Ansible Runner
- Ansible Collections
- Python libraries
- System dependencies
- Custom user needs

You can either use the default base execution environment included in your Ansible Automation Platform subscription, or you can define and create an automation execution environment using Ansible Builder.

3.8.1. Using the base automation execution environment

Your subscription with Ansible Automation Platform gives you access to some base automation execution environments. You can use a base automation execution environments as a starting point for creating a customized execution environment.

Base images included with Ansible Automation Platform are hosted on the Red Hat Ecosystem Catalog (registry.redhat.io).

Prerequisites

- You have a valid Red Hat Ansible Automation Platform subscription.

Procedure

1. Log in to registry.redhat.io.

```
$ podman login registry.redhat.io
```

2. Pull the base images from the registry:

```
$podman pull registry.redhat.io/aap/<image name>
```

3.8.1.1. Customize the base execution environment image

Ansible Automation Platform includes the following default execution environments:

- **Minimal** - Includes the latest Ansible-core 2.15 release along with Ansible Runner, but does not include collections or other content
- **EE Supported** - Minimal, plus all Red Hat-supported collections and dependencies

While these environments cover many automation use cases, you can add additional items to customize these containers for your specific needs. For more information about customizing your execution environment, see [Customizing an existing automation execution environment image](#) in the Creating and using execution environments guide.

3.8.1.2. About Ansible Builder

You also have the option of creating an entirely new execution environment with Ansible Builder, also referred to as execution environment builder. Ansible Builder is a command line tool you can use to create an execution environment for Ansible. You can only create execution environments with Ansible Builder.

To build your own execution environment, you must:

- Download Ansible Builder
- Create a definition file that defines your execution environment
- Create an execution environment image based on the definition file

For more information about building an execution environment, see [Creating and using execution environments](#).

3.8.2. Adding an execution environment to a job template

Prerequisites

- An execution environment must have been created using ansible-builder as described in [Using Ansible Builder](#). When an execution environment has been created, you can use it to run jobs. Use the automation controller UI to specify the execution environment to use in your job templates.
- Depending on whether an execution environment is made available for global use or tied to an organization, you must have the appropriate level of administrator privileges to use an execution environment in a job. Execution environments tied to an organization require Organization administrators to be able to run jobs with those execution environment.
- Before running a job or job template that uses an execution environment that has a credential assigned to it, ensure that the credential has a username, host, and password.

Procedure

1. From the navigation panel, select **Automation Execution → Infrastructure → Execution Environments**.
2. Click **Create execution environment** to create an execution environment.
3. Enter the appropriate details into the following fields:
 - a. **Name** (required): Enter a name for the execution environment.
 - b. **Image** (required): Enter the image name. The image name requires its full location (repository), the registry, image name, and version tag, as in the following example: **quay.io/ansible/awx-ee:latestrepo/project/image-name:tag**
 - c. Optional: **Pull**: Choose the type of pull when running jobs:
 - i. **Always pull container before running** Pulls the latest image file for the container.
 - ii. **Only pull the image if not present before running** Only pulls the latest image if none are specified.
 - iii. **Never pull container before running** Never pull the latest version of the container image.



NOTE

If you do not set a type for pull, the value defaults to **Only pull the image if not present before running**.

- d. Optional: **Description**: Enter an optional description.
 - e. Optional: **Organization**: Assign the organization to specifically use this execution environment. To make the execution environment available for use across multiple organizations, leave this field blank.
 - f. **Registry credential**: If the image has a protected container registry, provide the credential to access it.
4. Click **Create execution environment**. Your newly added execution environment is ready to be used in a job template.

5. To add an execution environment to a job template, navigate to **Automation Execution → Templates** and select your template. ..Click **Edit template** and specify your execution environment in the field labeled **execution environment**.

When you have added an execution environment to a job template, the template will be listed in the **Templates** tab in your execution environment details.

3.8.2.1. About container registries

If you have many execution environment that you want to maintain, you can store them in a container registry linked to your private automation hub.

For more information, see [Populating your private automation hub container registry](#) from the Creating and using execution environments guide.

3.9. BUILD AND USE A DECISION ENVIRONMENT

Event-Driven Ansible includes an `ansible.eda` collection, which contains sample sources, event filters and rulebooks. All the collections, ansible rulebooks and their dependencies use a decision environment, which is an image that can be run on either Podman or Kubernetes.

In decision environments, sources, which are typically Python code, are distributed through `ansible-collections`. They inject external events into a rulebook for processing. The rulebook consists of the following:

- The python interpreter
- Java Runtime Environment for Drools rule engine
- `ansible-rulebook` python package
- `ansible.eda` collection

You can use the base decision environment and build your own customized Decision Environments with additional collections and collection dependencies. You can build a decision environment using a Dockerfile or optionally you can deploy your CA certificate into the image.

3.9.1. Setting up a new decision environment

The following steps describe how to import a decision environment into the platform.

Prerequisites

- You have set up any necessary credentials. For more information, see the [Setting up credentials](#) section of the Using automation decisions guide.
- You have pushed a decision environment image to an image repository or you chose to use the image **de-supported** provided at registry.redhat.io.

Procedure

1. Navigate to **Automation Decisions → Decision Environments**.
2. Click **Create decision environment**.

3. Enter the following:

Name

Insert the name.

Description

This field is optional.

Image

This is the full image location, including the container registry, image name, and version tag.

Credential

This field is optional. This is the token needed to use the decision environment image.

4. Select **Create decision environment**.

Your decision environment is now created and can be managed on the **Decision Environments** page.

After saving the new decision environment, the decision environment's details page is displayed. From there or the **Decision Environments** list view, you can edit or delete it.

3.10. CREATING AN AUTOMATION EXECUTION PROJECT

A project is a logical collection of playbooks. Projects are useful as a way to group your automation content according to the organizing principle of your choice.

You can set up an automation execution project in the platform UI.

Procedure

Procedure . From the navigation panel, select **Automation Execution** → **Projects**. . On the **Projects** page, click **Create project** to launch the **Create Project** window.

+

1. Enter the appropriate details into the following required fields:

- **Name** (required)
- Optional: **Description**
- **Organization** (required): A project must have at least one organization. Select one organization now to create the project. When the project is created you can add additional organizations.
- Optional: **Execution Environment**: Enter the name of the execution environment or search from a list of existing ones to run this project. For more information, see [Migrating to automation execution environments](#) in the Upgrade and migration.
- **Source Control Type** (required): Select an SCM type associated with this project from the menu. Options in the following sections become available depending on the type chosen. For more information, see [Managing playbooks manually](#) or [Managing playbooks using source control](#).
- Optional: **Content Signature Validation Credential** Use this field to enable content verification. Specify the GPG key to use for validating content signature during project synchronization. If the content has been tampered with, the job will not run. For more

information, see [Project signing and verification](#).

2. Click **Create project**.

3.11. CREATING AN AUTOMATION DECISION PROJECT

Like automation execution projects, automation decision projects are logical collections of automation decision content. You can use the project function to organize your automation decision content in a way that makes sense to you.

Prerequisites

- You have set up any necessary credentials. For more information, see the [Setting up credentials](#) section of the Using automation decisions guide.
- You have an existing repository containing rulebooks that are integrated with playbooks contained in a repository to be used by automation controller.

Procedure

1. From the navigation panel, select **Automation Decisions → Projects**.
2. Click **Create project**.
3. Enter the following information:
 - **Name:** Enter project name.
 - **Description:** This field is optional.
 - **Organization:** Select the organization associated with the project.
 - **Source control type:** Git is the only SCM type available for use.
 - **Proxy:** Proxy used to access HTTP or HTTPS servers.
 - **Source control branch/tag/commit:** Branch to checkout. Can also be tags, commit hashes, or arbitrary refs.
 - **Source control refspec:** A refspec to fetch. This parameter allows access to references through the branch field not otherwise available.
 - Optional: **Source control credential:** The token needed to use the source control URL.
 - **Content signature validation credential:** Enables content signing to verify that the content has remained secure during project syncing. If the content is tampered with, the job will not run.
 - **Options:** Checking the box next to **Verify SSL** verifies the SSL with HTTPS when the project is imported.
4. Click **Create project**.

Your project is now created and can be managed in the **Projects** screen.

After saving the new project, the project's details page is displayed. From there or the **Projects** list view, you can edit or delete it.

3.12. ABOUT INVENTORIES

An inventory is a file listing the collection of hosts managed by Ansible Automation Platform. Organizations are assigned to inventories, while permissions to launch playbooks against inventories are controlled at the user or team level.

3.12.1. Browsing and creating inventories

You can find inventories in the UI by navigating to **Automation Execution** → **Infrastructure** → **Inventories**. The Inventories window displays a list of the inventories that are currently available. You can sort the inventory list by name and search by inventory type, organization, description, inventory creators or modifiers, or additional criteria. Use the following procedure to create a new inventory.

Procedure

1. From the navigation panel, select **Automation Execution** → **Infrastructure** → **Inventories**. The **Inventories** view displays a list of the inventories currently available.
2. Click **Create inventory**, and from the list menu select the type of inventory you want to create.
3. Enter the appropriate details into the following fields:
 - **Name:** Enter a name for the inventory.
 - Optional: **Description:** Enter a description.
 - **Organization:** Choose among the available organizations.
 - Only applicable to Smart Inventories: **Smart Host Filter:** Filters are similar to tags in that tags are used to filter certain hosts that contain those names. Therefore, to populate this field, specify a tag that contains the hosts you want, not the hosts themselves. Filters are case-sensitive. For more information, see [Smart host filters](#) in the Using automation execution guide.
 - **Instance groups:** Select the instance group or groups for this inventory to run on. If the list is extensive, use the search to narrow the options. You can select multiple instance groups and sort them in the order that you want them run.
 - Optional: **Labels:** Add labels that describe this inventory, so they can be used to group and filter inventories and jobs.
 - Only applicable to constructed inventories: **Input inventories:** Specify the source inventories to include in this constructed inventory. Empty groups from input inventories are copied into the constructed inventory.
 - Optional and only applicable to constructed inventories: **Cache timeout (seconds):** Set the length of time you want the cache plugin data to timeout.
 - Only applicable to constructed inventories: **Verbosity:** Control the level of output that Ansible produces as the playbook executes related to inventory sources associated with constructed inventories. Select the verbosity from Normal to various Verbose or Debug settings. This only appears in the "details" report view.

- Verbose logging includes the output of all commands.
- Debug logging is exceedingly verbose and includes information on SSH operations that can be useful in certain support instances. Most users do not need to see debug mode output.
- Only applicable to constructed inventories: **Limit**: Restricts the number of returned hosts for the inventory source associated with the constructed inventory. You can paste a group name into the limit field to only include hosts in that group. For more information, see the **Source variables** setting.
- Only applicable to standard inventories: **Options**: Check the box next to **Prevent instance group fallback** to enable only the instance groups listed in the **Instance groups** field to execute the job. If unchecked, all available instances in the execution pool will be used based on the hierarchy described in [Control where a job runs](#) in the Configuring automation execution guide. Click the tooltip for more information.



NOTE

Set the **prevent_instance_group_fallback** option for smart inventories through the API.

- **Variables** (**Source variables** for constructed inventories):
 - **Variables**: Variable definitions and values to apply to all hosts in this inventory. Enter variables using either JSON or YAML syntax. Use the radio button to toggle between the two.
 - **Source variables** for constructed inventories are used to configure the constructed inventory plugin. Source variables create groups under the **groups** data key. The variable accepts Jinja2 template syntax, renders it for every host, makes a **true** or **false** evaluation, and includes the host in the group (from the key of the entry) if the result is **true**.

4. Click **Create inventory**.

After creating the new inventory, you can proceed with configuring permissions, groups, hosts, sources, and viewing completed jobs, if applicable to the type of inventory.

3.12.2. Browsing and creating inventories

The Inventories window displays a list of the inventories that are currently available. You can sort the inventory list by name and searched type, organization, description, owners and modifiers of the inventory, or additional criteria.

Procedure

1. From the navigational panel, select **Automation Execution** → **Infrastructure** → **Inventories**.
2. Click **Create inventory**, and select the type of inventory to create.
3. Enter the appropriate details into the following fields:
 - **Name**: Enter a name appropriate for this inventory.
 - Optional: **Description**: Enter an arbitrary description as appropriate.

- **Organization:** Required. Choose among the available organizations.
- Only applicable to Smart Inventories: **Smart Host Filter:** Populate the hosts for this inventory by using a search filter.

Example


```
name__icontains=RedHat.
```

These options are based on the organization you chose.

Filters are similar to tags in that tags are used to filter certain hosts that contain those names. Therefore, to populate the **Smart Host Filter** field, specify a tag that contains the hosts you want, not the hosts themselves.

Filters are case-sensitive.

- **Instance Groups:** Select the instance group or groups for this inventory to run on. You can select many instance groups and sort them in the order that you want them run.
- Optional: **Labels:** Supply labels that describe this inventory, so they can be used to group and filter inventories and jobs.
- Only applicable to constructed inventories: **Input inventories:** Specify the source inventories to include in this constructed inventory. Empty groups from input inventories are copied into the constructed inventory.
- Optional:(Only applicable to constructed inventories): **Cached timeout (seconds):** Set the length of time you want the cache plugin data to timeout.
- Only applicable to constructed inventories: **Verbosity:** Control the level of output that Ansible produces as the playbook executes related to inventory sources associated with constructed inventories.
Select the verbosity from:
 - **Normal**
 - **Verbose**
 - **More verbose**
 - **Debug**
 - **Connection Debug**
 - **WinRM Debug**
 - **Verbose** logging includes the output of all commands.
 - **More verbose** provides more detail than **Verbose**.
 - **Debug** logging is exceedingly verbose and includes information about SSH operations that can be useful in certain support instances. Most users do not need to see debug mode output.
 - **Connection Debug** enables you to run ssh in verbose mode, providing debugging information about the SSH connection progress.
 - **WinRM Debug** used for verbosity specific to windows remote management

Click the  icon for information on **How to use the constructed inventory plugin**

- Only applicable to constructed inventories: **Limit**: Restricts the number of returned hosts for the inventory source associated with the constructed inventory. You can paste a group name into the limit field to only include hosts in that group. For more information, see the **Source vars** setting.
- Only applicable to standard inventories: **Options**: Check the **Prevent Instance Group Fallback** option to enable only the instance groups listed in the **Instance Groups** field to execute the job. If unchecked, all available instances in the execution pool are used based on the hierarchy.
- **Variables** (**Source vars** for constructed inventories):
 - **Variables** Variable definitions and values to apply to all hosts in this inventory. Enter variables by using either JSON or YAML syntax. Use the radio button to toggle between the two.
 - **Source vars** for constructed inventories creates groups, specifically under the **groups** key of the data. It accepts Jinja2 template syntax, renders it for every host, makes a **true** or **false** evaluation, and includes the host in the group (from the key of the entry) if the result is **true**. This is particularly useful because you can paste that group name into the limit field to only include hosts in that group.

4. Click **Create inventory**.

After saving the new inventory, you can proceed with configuring permissions, groups, hosts, sources, and view completed jobs, if applicable to the type of inventory.

3.13. WORK WITH JOB TEMPLATES

A job template is a definition and set of parameters for running an Ansible job.

A job template combines an Ansible playbook from a project and the settings required to launch it, including information about the target host against which the playbook will run, authentication information to access the host, and any other relevant variables. Job templates are useful to run the same job many times. Job templates also encourage the reuse of Ansible playbook content and collaboration between teams. For more information, see Job Templates in the Automation controller User Guide.

3.13.1. Getting started with job templates

As part of the initial setup, a **Demo Job Template** is created for you.

Procedure

1. To review existing templates, select **Automation Execution** → **Templates** from the navigation panel.
2. Click **Demo Job Template** to view its details.

3.13.2. Creating a job template

Procedure

1. From the navigation panel, select **Automation Execution → Templates**.
2. On the **Templates** page, select **Create job template** from the **Create template** list.
3. Enter the appropriate details in the following fields:



NOTE


If a field has the **Prompt on launch** checkbox selected, launching the job prompts you for the value for that field when launching.

Most prompted values override any values set in the job template.

Exceptions are noted in the following table.


Field	Options	Prompt on Launch
Name	Enter a name for the job.	N/A
Description	Enter an arbitrary description as appropriate (optional).	N/A
Job Type	<p>Choose a job type:</p> <ul style="list-style-type: none"> ● Run: Start the playbook when launched, running Ansible tasks on the selected hosts. ● Check: Perform a "dry run" of the playbook and report changes that would be made without actually making them. Tasks that do not support check mode are missed and do not report potential changes. <p>For more information about job types see the Playbooks section of the Ansible documentation.</p>	Yes
Inventory	<p>Choose the inventory to use with this job template from the inventories available to the logged in user.</p> <p>A System Administrator must grant you or your team permissions to be able to use certain inventories in a job template.</p>	<p>Yes.</p> <p>Inventory prompts show up as its own step in a later prompt window.</p>

Field	Options	Prompt on Launch
Project	Select the project to use with this job template from the projects available to the user that is logged in.	N/A
Source control branch	<p>This field is only present if you chose a project that allows branch override. Specify the overriding branch to use in your job run. If left blank, the specified SCM branch (or commit hash or tag) from the project is used.</p> <p>For more information, see Job branch overriding.</p>	Yes
Execution Environment	Select the container image to be used to run this job. You must select a project before you can select an execution environment.	<p>Yes.</p> <p>Execution environment prompts show up as its own step in a later prompt window.</p>
Playbook	Choose the playbook to be launched with this job template from the available playbooks. This field automatically populates with the names of the playbooks found in the project base path for the selected project. Alternatively, you can enter the name of the playbook if it is not listed, such as the name of a file (such as foo.yml) you want to use to run with that playbook. If you enter a filename that is not valid, the template displays an error, or causes the job to fail.	N/A

Field	Options	Prompt on Launch
<p>Credentials</p>	<p>Select the  icon to open a separate window.</p> <p>Choose the credential from the available options to use with this job template.</p> <p>Use the drop-down menu list to filter by credential type if the list is extensive. Some credential types are not listed because they do not apply to certain job templates.</p>	<ul style="list-style-type: none"> ● If selected, when launching a job template that has a default credential and supplying another credential replaces the default credential if it is the same type. The following is an example this message: <p>Job Template default credentials must be replaced with one of the same type. Please select a credential for the following types in order to proceed: Machine.</p> <ul style="list-style-type: none"> ● You can add more credentials as you see fit. ● Credential prompts show up as its own step in a later prompt window.

Field	Options	Prompt on Launch
Labels	<ul style="list-style-type: none"> Optionally supply labels that describe this job template, such as dev or test. Use labels to group and filter job templates and completed jobs in the display. Labels are created when they are added to the job template. Labels are associated with a single Organization by using the Project that is provided in the job template. Members of the Organization can create labels on a job template if they have edit permissions (such as the admin role). Once you save the job template, the labels appear in the Job Templates overview in the Expanded view. Select X beside a label to remove it. When a label is removed, it is no longer associated with that particular Job or Job Template, but it remains associated with any other jobs that reference it. Jobs inherit labels from the Job Template at the time of launch. If you delete a label from a Job Template, it is also deleted from the Job. 	<ul style="list-style-type: none"> If selected, even if a default value is supplied, you are prompted when launching to supply additional labels, if needed. You cannot delete existing labels, selecting X only removes the newly added labels, not existing default labels.
Forks	The number of parallel or simultaneous processes to use while executing the playbook. A value of zero uses the Ansible default setting, which is five parallel processes unless overridden in /etc/ansible/ansible.cfg .	Yes

Field	Options	Prompt on Launch
Limit	<p>A host pattern to further constrain the list of hosts managed or affected by the playbook. You can separate many patterns by colons (:). As with core Ansible:</p> <ul style="list-style-type: none"> ● a:b means "in group a or b" ● a:b&c means "in a or b but must be in c" ● a:!b means "in a, and definitely not in b" <p>For more information, see Patterns: targeting hosts and groups in the Ansible documentation.</p>	<p>Yes</p> <p>If not selected, the job template executes against all nodes in the inventory or only the nodes predefined on the Limit field. When running as part of a workflow, the workflow job template limit is used instead.</p>
Verbosity	<p>Control the level of output Ansible produces as the playbook executes. Choose the verbosity from Normal to various Verbose or Debug settings. This only appears in the details report view.</p> <p>Verbose logging includes the output of all commands. Debug logging is exceedingly verbose and includes information about SSH operations that can be useful in certain support instances.</p> <p>Verbosity 5 causes automation controller to block heavily when jobs are running, which could delay reporting that the job has finished (even though it has) and can cause the browser tab to lock up.</p>	<p>Yes</p>
Job Slicing	<p>Specify the number of slices you want this job template to run. Each slice runs the same tasks against a part of the inventory. For more information about job slices, see Job Slicing.</p>	<p>Yes</p>

Field	Options	Prompt on Launch
Timeout	<p>This enables you to specify the length of time (in seconds) that the job can run before it is canceled. Consider the following for setting the timeout value:</p> <ul style="list-style-type: none"> • There is a global timeout defined in the settings which defaults to 0, indicating no timeout. • A negative timeout (<0) on a job template is a true "no timeout" on the job. • A timeout of 0 on a job template defaults the job to the global timeout (which is no timeout by default). • A positive timeout sets the timeout for that job template. 	Yes
Show Changes	Enables you to see the changes made by Ansible tasks.	Yes
Instance Groups	<p>Choose Instance and Container Groups to associate with this job template. If the list is extensive, use the  icon to narrow the options. Job template instance groups contribute to the job scheduling criteria, see Job Runtime Behavior and Control where a job runs for rules. A System Administrator must grant you or your team permissions to be able to use an instance group in a job template. Use of a container group requires admin rights.</p>	<ul style="list-style-type: none"> • Yes. <p>If selected, you are providing the jobs preferred instance groups in order of preference. If the first group is out of capacity, later groups in the list are considered until one with capacity is available, at which point that is selected to run the job.</p> <ul style="list-style-type: none"> • If you prompt for an instance group, what you enter replaces the normal instance group hierarchy and overrides all of the organizations' and inventories' instance groups. • The Instance Groups prompt shows up as its own step in a later prompt window.

Field	Options	Prompt on Launch
Job Tags	Type and select the Create menu to specify which parts of the playbook should be executed. For more information and examples see Tags in the Ansible documentation.	Yes
Skip Tags	Type and select the Create menu to specify certain tasks or parts of the playbook to skip. For more information and examples see Tags in the Ansible documentation.	Yes
Extra Variables	<ul style="list-style-type: none"> ● Pass extra command line variables to the playbook. This is the "-e" or "-extra-vars" command line parameter for <code>ansible-playbook</code> that is documented in the Ansible documentation at Defining variables at runtime. ● Provide key or value pairs by using either YAML or JSON. These variables have a maximum value of precedence and overrides other variables specified elsewhere. The following is an example value: git_branch: production release_version: 1.5 	<p>Yes.</p> <p>If you want to be able to specify extra_vars on a schedule, you must select Prompt on launch for Variables on the job template, or enable a survey on the job template. Those answered survey questions become extra_vars.</p>

4. You can set the following options for launching this template, if necessary:

- **Privilege escalation:** If checked, you enable this playbook to run as an administrator. This is the equal of passing the **--become** option to the **ansible-playbook** command.
- **Provisioning callback:** If checked, you enable a host to call back to automation controller through the REST API and start a job from this job template. For more information, see [Provisioning Callbacks](#).
- **Enable webhook:** If checked, you turn on the ability to interface with a predefined SCM system web service that is used to launch a job template. GitHub and GitLab are the supported SCM systems.
 - If you enable webhooks, other fields display, prompting for additional information:
 - **Webhook service:** Select which service to listen for webhooks from.

- **Webhook URL:** Automatically populated with the URL for the webhook service to POST requests to.
 - **Webhook key:** Generated shared secret to be used by the webhook service to sign payloads sent to automation controller. You must configure this in the settings on the webhook service in order for automation controller to accept webhooks from this service.
 - **Webhook credential:** Optionally, give a GitHub or GitLab personal access token (PAT) as a credential to use to send status updates back to the webhook service. Before you can select it, the credential must exist.

See [Credential Types](#) to create one.
 - For additional information about setting up webhooks, see [Working with Webhooks](#).
 - **Concurrent jobs:** If checked, you are allowing jobs in the queue to run simultaneously if not dependent on one another. Check this box if you want to run job slices simultaneously. For more information, see [Automation controller capacity determination and job impact](#).
 - **Enable fact storage:** If checked, automation controller stores gathered facts for all hosts in an inventory related to the job running.
 - **Prevent instance group fallback** Check this option to allow only the instance groups listed in the **Instance Groups** field to run the job. If clear, all available instances in the execution pool are used based on the hierarchy described in [Control where a job runs](#).
5. Click **Create job template**, when you have completed configuring the details of the job template.

Creating the template does not exit the job template page but advances to the Job Template **Details** tab. After saving the template, you can click **Launch template** to start the job. You can also click **Edit** to add or change the attributes of the template, such as permissions, notifications, view completed jobs, and add a survey (if the job type is not a scan). You must first save the template before launching, otherwise, **Launch template** remains disabled.

Verification

1. From the navigation panel, select **Automation Execution → Templates**.
2. Verify that the newly created template appears on the **Templates** page.

3.13.3. Running a job template

One benefit of automation controller is the push-button deployment of Ansible playbooks. You can configure a template to store all the parameters that you would normally pass to the Ansible playbook on the command line. In addition to the playbooks, the template passes the inventory, credentials, extra variables, and all options and settings that you can specify on the command line.

3.13.4. Editing a job template

As part of the initial setup, you can leave the default **Demo Job Template** as it is, but you can edit it later.

Procedure

1. Open the template to edit it by using one of these methods:
 - Click **Edit** in the job template Details page.
 - From the navigation panel, select **Automation Execution** → **Templates**. Click **Edit** next to the template name and edit the appropriate details.
2. Save your changes.

Templates > Demo Job Template ⌵

Edit Details

Name *

Description

Job Type * ⌵ Prompt on launch

Run

Inventory * ⌵ Prompt on launch

Project * ⌵

Execution Environment ⌵

Playbook * ⌵

hello_world.yml

Credentials ⌵ Prompt on launch

Labels ⌵

Variables ⌵ YAML JSON Prompt on launch ⌵

1 ---

2

Forks ⌵

Limit ⌵ Prompt on launch

Verbosity ⌵ Prompt on launch

0 (Normal)

Job Slicing ⌵

Timeout ⌵

Show Changes ⌵ Prompt on launch

Off

Instance Groups ⌵

Job Tags ⌵ Prompt on launch

Skip Tags ⌵ Prompt on launch

Options

Privilege Escalation ⌵ Provisioning Callbacks ⌵ Enable Webhook ⌵ Concurrent Jobs ⌵ Enable Fact Storage ⌵

Save
Cancel

3. To exit after saving and return to the **Templates** list view, use the breadcrumb navigation links or click **Cancel**. Clicking **Save** does not exit the **Details** dialog.

3.14. CREATE AND RUN A RULEBOOK ACTIVATION

In Event-Driven Ansible, a rulebook activation is a process running in the background defined by a decision environment executing a specific rulebook.

3.14.1. Setting up a rulebook activation

Prerequisites

- You have set up a project.

- You have set up a decision environment.
- You have set up an automation controller token. See [Setting up an automation controller token](#) in the Using automation decisions guide for more information.

Procedure

1. From the navigation panel, select **Automation Decisions** → **Rulebook Activations**.
2. Click **Create rulebook activation**.
3. Enter the following information:
 - **Name:** Insert the name.
 - **Description:** This field is optional.
 - **Organization:** This field is optional.
 - **Project:** This field is optional.
 - **Rulebook:** Rulebooks are displayed according to the project you selected.
 - **Credential:** Select 0 or more credentials for this rulebook activation. This field is optional.



NOTE

The credentials that display in this field are customized based on your rulebook activation and only include the following credential types: Vault, Red Hat Ansible Automation Platform, or any custom credential types that you have created. For more information about credentials, see [Setting up credentials for Event-Driven Ansible controller](#) in the Using automation execution guide.

- **Decision environment:** A decision environment is a container image to run Ansible rulebooks.



NOTE

In Event-Driven Ansible controller, you cannot customize the pull policy of the decision environment. By default, it follows the behavior of the **always** policy. Every time an activation is started, the system tries to pull the most recent version of the image.

- **Restart policy.** This is the policy that determines how an activation should restart after the container process running the source plugin ends. Select from the following options:
 - **Always:** This restarts the rulebook activation immediately, regardless of whether it ends successfully or not, and occurs no more than 5 times.
 - **Never:** This never restarts a rulebook activation when the container process ends.
 - **On failure:** This restarts the rulebook activation after 60 seconds by default, only when the container process fails, and occurs no more than 5 times.

- **Log level:** This field defines the severity and type of content in your logged events. Select from one of the following options:
 - **Error:** Logs that contain error messages that are displayed in the **History** tab of an activation.
 - **Info:** Logs that contain useful information about rulebook activations, such as a success or failure, triggered action names and their related action events, and errors.
 - **Debug:** Logs that contain information that is only useful during the debug phase and might be of little value during production. This log level includes both error and log level data.
- **Service name:** This defines a service name for Kubernetes to configure inbound connections if the activation exposes a port. This field is optional.
- **Rulebook activation enabled?:** Toggle to automatically enable the rulebook activation to run.
- **Variables:** The variables for the rulebook are in JSON or YAML format. The content would be equivalent to the file passed through the **--vars** flag of `ansible-rulebook` command.
- **Options:** Check the **Skip audit events** option if you do not want to see your events in the Rule Audit.

4. Click **Create rulebook activation**.

Your rulebook activation is now created and can be managed on the **Rulebook Activations** page.

After saving the new rulebook activation, the rulebook activation's details page is displayed, with either a **Pending**, **Running**, or **Failed** status. From there or the **Rulebook Activations** list view, you can restart or delete it.



NOTE

Occasionally, when a source plugin shuts down, it causes a rulebook to exit gracefully after a certain amount of time. When a rulebook activation shuts down, any tasks that are waiting to be performed will be canceled, and an info level message is sent to the activation log. For more information, see [Rulebooks](#).

3.14.1.1. Rulebook activation list view

On the **Rulebook Activations** page, you can view the rulebook activations that you have created along with the **Status**, **Number of rules** with the rulebook, the **Fire count**, and **Restart count**.

If the **Status** is **Running**, it means that the rulebook activation is running in the background and executing the required actions according to the rules declared in the rulebook.

You can view more details by selecting the activation from the **Rulebook Activations** list view.

The screenshot shows the 'Rulebook Activations' page in the Ansible Automation Platform. The page title is 'Rulebook Activations' and it includes a subtitle: 'Rulebook activations are rulebooks that have been activated to run.' Below the title is a search bar for 'Name' and a '+ Create rulebook activation' button. The main content is a table with the following data:

Name	Activation status	Number of rules	Fire count	Restart count	
Activation 3	Failed	0	0	0	Toggle on, More Actions
Activation 2	Failed	0	0	0	Toggle on, More Actions
Activation 1	Running	1	6	4	Toggle on, More Actions

For all activations that have run, you can view the **Details** and **History** tabs to get more information about what happened.

3.14.2. Enabling and disabling rulebook activations

Procedure

1. Select the switch on the row level to enable or disable your chosen rulebook.
2. In the window, select **Yes, I confirm that I want to enable/disable these X rulebook activations.**
3. Select **Enable/Disable rulebook activation.**

3.14.3. Restarting rulebook activations



NOTE


You can only restart a rulebook activation if it is currently enabled and the restart policy was set to **Always** when it was created.

Procedure

1. Select the **More Actions** icon **⋮** next to **Rulebook Activation enabled/disabled** toggle.
2. Select **Restart rulebook activation.**
3. In the window, select **Yes, I confirm that I want to restart these X rulebook activations.**
4. Select **Restart rulebook activations.**

3.14.4. Deleting rulebook activations

Procedure

1. Select the **More Actions** icon  next to the **Rulebook Activation enabled/disabled** toggle.
2. Select **Delete rulebook activation**.
3. In the window, select **Yes, I confirm that I want to delete these X rulebook activations**.
4. Select **Delete rulebook activations**.

3.14.5. Activating webhook rulebooks

In Openshift environments, you can allow webhooks to reach an activation-job-pod over a given port by creating a Route that exposes that rulebook activation's Kubernetes service.

Prerequisites

- You have created a rulebook activation.



NOTE

The following is an example of rulebook with a given webhook:

```
- name: Listen for storage-monitor events
  hosts: all
  sources:
    - ansible.eda.webhook:
        host: 0.0.0.0
        port: 5000
  rules:
    - name: Rule - Print event information
      condition: event.meta.headers is defined
      action:
        run_job_template:
          name: StorageRemediation
          organization: Default
          job_args:
            extra_vars:
              message: from eda
              sleep: 1
```

Procedure

1. Create a Route (on OpenShift Container Platform) to expose the service. The following is an example Route for an ansible-rulebook source that expects POST's on port 5000 on the decision environment pod:

```
kind: Route
apiVersion: route.openshift.io/v1
metadata:
  name: test-sync-bug
  namespace: dynatrace
labels:
  app: eda
  job-name: activation-job-1-5000
spec:
```

```
host: test-sync-bug-dynatrace.apps.aap-dt.ocp4.testing.ansible.com
to:
  kind: Service
  name: activation-job-1-5000
  weight: 100
port:
  targetPort: 5000
tls:
  termination: edge
  insecureEdgeTerminationPolicy: Redirect
  wildcardPolicy: None
```

2. When you create the Route, test it with a **Post to the Route URL**:

```
curl -H "Content-Type: application/json" -X POST
test-sync-bug-dynatrace.apps.aap-dt.ocp4.testing.ansible.com -d
'{}'
```



NOTE

You do not need the port as it is specified on the Route (targetPort).

CHAPTER 4. GETTING STARTED AS AN AUTOMATION OPERATOR

As an automation operator, Ansible Automation Platform can help you organize and manage automation projects using Red Hat certified collections or custom content for your organization.

To get started as a platform operator, see the following sections:

- [Get started with playbooks](#)
- [Publishing to a collection in a source code manager](#)
- [Automation execution projects](#)
- [Build and use an execution environment](#)
- [Job templates](#)
- [About inventories](#)
- [Automation execution jobs](#)

4.1. GET STARTED WITH PLAYBOOKS

A playbook runs tasks in order from top to bottom. Within each play, tasks also run in order from top to bottom.

4.1.1. Learn about playbooks

Playbooks with multiple “plays” can orchestrate multi-machine deployments, running one play on your web servers, another play on your database servers, and a third play on your network infrastructure.

For more information, see [Getting started with playbooks](#).

4.2. WRITING A PLAYBOOK

Create a playbook that pings your hosts and prints a "Hello world" message.

Ansible uses the YAML syntax. YAML is a human-readable language that enables you to create playbooks without having to learn a complicated coding language.

Procedure

1. Create a file named **playbook.yaml** in your **ansible_quickstart** directory, with the following content:

```
- name: My first play
  hosts: myhosts
  tasks:
    - name: Ping my hosts
      ansible.builtin.ping:
```

```
- name: Print message
  ansible.builtin.debug:
    msg: Hello world
```

2. Run your playbook:

```
$ ansible-playbook -i inventory.ini playbook.yaml
```

Ansible returns the following output:

```
PLAY [My first play] *****
TASK [Gathering Facts] *****
ok: [192.0.2.50]
ok: [192.0.2.51]
ok: [192.0.2.52]

TASK [Ping my hosts] *****
ok: [192.0.2.50]
ok: [192.0.2.51]
ok: [192.0.2.52]

TASK [Print message] *****
ok: [192.0.2.50] => {
  "msg": "Hello world"
}
ok: [192.0.2.51] => {
  "msg": "Hello world"
}
ok: [192.0.2.52] => {
  "msg": "Hello world"
}

PLAY RECAP *****
192.0.2.50: ok=3  changed=0  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
192.0.2.51: ok=3  changed=0  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
192.0.2.52: ok=3  changed=0  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
```

Additional resources

- For more information on playbooks, see [Getting Started with Ansible Playbooks](#).
- If you need help writing a playbook, see [Red Hat Ansible Lightspeed with IBM watsonx Code Assistant](#).

4.3. BUNDLE CONTENT WITH ANSIBLE ROLES

A role is like a customized piece of automation content that bundles together relevant bits from playbooks to fit your system's specific needs. Roles are self-contained and portable, and can include groupings of tasks, variables, configuration templates, handlers, and other supporting files to orchestrate complicated automation flows.

Instead of creating huge playbooks with hundreds of tasks, you can use roles to break the tasks apart into smaller, more discrete units of work.

To learn more about roles, see [What is an Ansible Role—and how is it used?](#) .

4.3.1. Creating a role

You can create roles using the Ansible Galaxy CLI tool, which is included with your Ansible Automation Platform bundle. Access role-specific commands from the `roles` subcommand:

```
ansible-galaxy role init <role_name>
```

Standalone roles outside of collections are still supported, but we recommend that you create new roles inside of a collection to take advantage of all the features Ansible Automation Platform has to offer.

4.3.2. Creating a role

You can create roles using the Ansible Galaxy CLI tool, which is included with your Ansible Automation Platform bundle. Access role-specific commands from the `role` subcommand:

Standalone roles outside of Collections are supported. Create new roles inside a Collection to take advantage of the features Ansible Automation Platform has to offer.

Procedure

1. In a terminal, navigate to the `roles` directory inside a collection.
2. Create a role called `my_role` inside the collection:

```
$ ansible-galaxy role init my_role
```

The collection now includes a role named `my_role` inside the `roles` directory, as you can see in this example:

```
~/.ansible/collections/ansible_collections/<my_namespace>/<my_collection_name>
...
└─ roles/
    └─ my_role/
        ├── .travis.yml
        ├── README.md
        ├── defaults/
        │   └─ main.yml
        ├── files/
        ├── handlers/
        │   └─ main.yml
        ├── meta/
        │   └─ main.yml
        ├── tasks/
        │   └─ main.yml
        ├── templates/
        ├── tests/
        │   ├── inventory
        │   └─ test.yml
        └─ vars/
            └─ main.yml
```

-
- 3. A custom role skeleton directory can be supplied by using the **--role-skeleton** argument. This allows organizations to create standardized templates for new roles to suit their needs.

```
$ ansible-galaxy role init my_role --role-skeleton ~/role_skeleton
```

This creates a role named **my_role** by copying the contents of `~/role_skeleton` into **my_role**. The contents of **role_skeleton** can be any files or folders that are valid inside a role directory.

For more information on creating roles, see [Aine's new devtools content].

4.4. ABOUT AUTOMATION CONTENT

Use the following Ansible concepts to create successful Ansible Playbooks and automation execution environments before beginning your Ansible development project.

4.4.1. About content collections

Ansible content collections are assemblages of automation content. There are two types of Ansible collections:

- **Ansible Certified Content Collections**, which contain fully-supported roles and modules that are enterprise- and production-ready for use in your environments.
- **Ansible validated content collections** which provide you with a trusted, expert-guided approach for performing foundational operations and tasks in your product.

Both types of content collections can be found in automation hub through the [Hybrid Cloud Console](#).

4.4.1.1. Downloading content

After collections are finalized, you can import them to a location where they can be distributed to others across your organization.

Procedure

1. Log in to Red Hat Ansible Automation Platform.
2. From the navigation panel, select **Automation Content** → **Collections**. The **Collections** page displays all collections across all repositories. You can search for a specific collection.
3. Select the collection that you want to export. The collection details page opens.
4. From the **Install** tab, select **Download tarball**. The .tar file is downloaded to your default browser downloads folder. You can now import it to the location of your choosing.

4.4.2. Browse content

Ansible Certified Content Collections are included in your subscription to Red Hat Ansible Automation Platform. Using Ansible automation hub, you can access and curate a unique set of collections from all forms of Ansible content.

Red Hat Ansible content contains two types of content:

- Ansible Certified Content Collections

- Ansible validated content collections

Ansible validated content collections are available in your private automation hub through the Platform Installer. When you download Ansible Automation Platform with the bundled installer, validated content is pre-populated into the private automation hub by default, but only if you enable the private automation hub as part of the inventory.

If you are not using the bundle installer, you can use a Red Hat-supplied Ansible Playbook to install validated content. For further information, see [Ansible validated content](#).

You can update validated collections manually by downloading their updated packages in automation hub.

4.4.3. Downloading content

After collections are finalized, you can import them to a location where they can be distributed to others across your organization.

Procedure

1. Log in to Red Hat Ansible Automation Platform.
2. From the navigation panel, select **Automation Content** → **Collections**. The **Collections** page displays all collections across all repositories. You can search for a specific collection.
3. Select the collection that you want to export. The collection details page opens.
4. From the **Install** tab, select **Download tarball**. The **.tar** file is downloaded to your default browser downloads folder. You can now import it to the location of your choosing.

4.5. PUBLISHING TO A COLLECTION

You can configure your projects to be uploaded to Git, or to the source control manager of your choice.

Procedure

1. From the navigation panel, select **Automation Execution** → **Projects**.
2. Locate or create the project that you want to publish to your source control manager.
3. In the project **Details** tab, select **Edit project**.
4. Select **Git** from the **Source Control Type** drop-down menu.
5. Enter the appropriate details into the following fields:
 - a. **Source Control URL** - see an example in the tooltip.
 - b. Optional: **Source control branch/tag/commit**. Enter the SCM branch, tags, commit hashes, arbitrary refs, or revision number (if applicable) from the source control to checkout. Some commit hashes and references might not be available unless you also provide a custom refs spec in the next field. If left blank, the default is **HEAD**, which is the last checked out branch, tag, or commit for this project.

- c. **Source Control Refspec**- This field is an option specific to Git source control and only advanced users familiar and comfortable with Git should specify which references to download from the remote repository. For more information, see [Job branch overriding](#).
 - d. **Source Control Credential**- If authentication is required, select the appropriate source control credential.
6. Optional: **Options** - select the launch behavior, if applicable:
 - a. **Clean** - Removes any local modifications before performing an update.
 - b. **Delete** - Deletes the local repository in its entirety before performing an update. Depending on the size of the repository this can significantly increase the amount of time required to complete an update.
 - c. **Track submodules** - Tracks the latest commit. See the tooltip for more information.
 - d. **Update Revision on Launch**- Updates the revision of the project to the current revision in the remote source control, and caches the roles directory from [Ansible Galaxy](#) or [Collections support](#). Automation controller ensures that the local revision matches and that the roles and collections are up-to-date with the last update. In addition, to avoid job overflows if jobs are spawned faster than the project can synchronize, selecting this enables you to configure a cache timeout to cache previous project synchronizations for a given number of seconds.
 - e. **Allow Branch Override** - Enables a job template or an inventory source that uses this project to start with a specified SCM branch or revision other than that of the project. For more information, see [Job branch overriding](#).
7. Click **Save** to save your project.

4.5.1. Manage collections in automation hub

As a platform operator, you can use namespaces in automation hub to curate and manage collections for the following purposes:

- Create groups with permissions to curate namespaces and upload collections to private automation hub.
- Add information and resources to the namespace to help end users of the collection in their automation tasks.
- Upload collections to the namespace.
- Review the namespace import logs to decide the success or failure of uploading the collection and its current approval status.

For more information about collections, see [Managing automation content](#).

4.5.2. Uploading a collection to automation hub

If you want to share a collection that you have created with the rest of the Ansible community, you can upload it to automation hub.



NOTE

Sharing a collection with the Ansible community requires getting the collection certified or validated by our Partner Engineering team. This action is available only to partner clients. For more about becoming a partner, see our [documentation on software certification](#).

You can upload your collection by using either the automation hub user interface or the **ansible-galaxy** client.

Prerequisites

- You have configured the **ansible-galaxy** client for automation hub.
- You have at least one namespace.
- You have run all content through **ansible-test sanity**

Procedure

1. From the navigation panel, select **Automation Content** → **Namespaces**.
2. Within the My namespaces tab, locate and click into the namespace to which you want to upload a collection.
3. Select the **Collections** tab, and then click **Upload collection**.
4. In the New collection modal, click **Select file**. Locate the file on your system.
5. Click **Upload**.

Using the **ansible-galaxy** client, enter the following command:

```
$ ansible-galaxy collection publish path/to/my_namespace-my_collection-1.0.0.tar.gz --api-key=SECRET
```

4.6. BUILD AND USE AN EXECUTION ENVIRONMENT

All automation in Red Hat Ansible Automation Platform runs on container images called automation execution environments.

Automation execution environments are consistent and shareable container images that serve as Ansible control nodes. Automation execution environments reduce the challenge of sharing Ansible content that has external dependencies. If automation content is like a script that a developer has written, an automation execution environment is like a replica of that developer's environment, thereby enabling you to reproduce and scale the automation content that the developer has written. In this way, execution environments make it easier for you to implement automation in a range of environments.

Automation execution environments contain:

- Ansible Core
- Ansible Runner
- Ansible Collections

- Python libraries
- System dependencies
- Custom user needs

You can either use the default base execution environment included in your Ansible Automation Platform subscription, or you can define and create an automation execution environment using Ansible Builder.

4.6.1. Using the base automation execution environment

Your subscription with Ansible Automation Platform gives you access to some base automation execution environments. You can use a base automation execution environments as a starting point for creating a customized execution environment.

Base images included with Ansible Automation Platform are hosted on the Red Hat Ecosystem Catalog (registry.redhat.io).

Prerequisites

- You have a valid Red Hat Ansible Automation Platform subscription.

Procedure

1. Log in to registry.redhat.io.

```
$ podman login registry.redhat.io
```

2. Pull the base images from the registry:

```
$podman pull registry.redhat.io/aap/<image name>
```

4.6.1.1. Customize the base execution environment image

Ansible Automation Platform includes the following default execution environments:

- **Minimal** - Includes the latest Ansible-core 2.15 release along with Ansible Runner, but does not include collections or other content
- **EE Supported** - Minimal, plus all Red Hat-supported collections and dependencies

While these environments cover many automation use cases, you can add additional items to customize these containers for your specific needs. For more information about customizing your execution environment, see [Customizing an existing automation execution environment image](#) in the Creating and using execution environments guide.

4.6.1.2. About Ansible Builder

You also have the option of creating an entirely new execution environment with Ansible Builder, also referred to as execution environment builder. Ansible Builder is a command line tool you can use to create an execution environment for Ansible. You can only create execution environments with Ansible Builder.

To build your own execution environment, you must:

- Download Ansible Builder
- Create a definition file that defines your execution environment
- Create an execution environment image based on the definition file

For more information about building an execution environment, see [Creating and using execution environments](#).

4.6.2. Adding an execution environment to a job template

Prerequisites

- An execution environment must have been created using ansible-builder as described in [Build an execution environment](#). When an execution environment has been created, you can use it to run jobs. Use the automation controller UI to specify the execution environment to use in your job templates.
- Depending on whether an execution environment is made available for global use or tied to an organization, you must have the appropriate level of administrator privileges to use an execution environment in a job. Execution environments tied to an organization require Organization administrators to be able to run jobs with those execution environments.
- Before running a job or job template that uses an execution environment that has a credential assigned to it, ensure that the credential contains a username, host, and password.

Procedure

1. From the navigation panel, select **Automation Execution → Infrastructure → Execution Environments**.
2. Click **Create execution environment** to add an execution environment.
3. Enter the appropriate details into the following fields:
 - **Name** (required): Enter a name for the execution environment.
 - **Image** (required): Enter the image name. The image name requires its full location (repository), the registry, image name, and version tag in the example format of **quay.io/ansible/awx-ee:latestrepo/project/image-name:tag**.
 - Optional: **Pull**: Choose the type of pull when running jobs:
 - **Always pull container before running** Pulls the latest image file for the container.
 - **Only pull the image if not present before running** Only pulls the latest image if none is specified.
 - **Never pull container before running** Never pull the latest version of the container image.



NOTE

If you do not set a type for pull, the value defaults to **Only pull the image if not present before running**.

- Optional: **Description**:
 - Optional: **Organization**: Assign the organization to specifically use this execution environment. To make the execution environment available for use across multiple organizations, leave this field blank.
 - **Registry Credential**: If the image has a protected container registry, give the credential to access it.
4. Click **Create execution environment**.
Your newly added execution environment is ready to be used in a job template.
 5. To add an execution environment to a job template, specify it in the **Execution Environment** field of the job template.

When you have added an execution environment to a job template, those templates are listed in the **Templates** tab of the execution environment:

4.7. AUTOMATION EXECUTION PROJECTS

A project is a logical collection of Ansible playbooks that you can manage in Ansible Automation Platform.

Platform administrators and automation developers have the permissions to create projects. As an automation operator you can view and sync projects.

4.7.1. Viewing project details

The **Projects** page displays a list of projects that are currently available.

1. From the navigation panel, select **Automation Execution** → **Projects**.
2. Click a project to view its details.
3. For each project listed, you can sync the latest revision, edit the project, or copy the project's attributes using the icons next to each project.

4.8. WORK WITH JOB TEMPLATES

A job template is a definition and set of parameters for running an Ansible job.

A job template combines an Ansible Playbook from a project with the settings required to launch the job. Job templates are useful for running the same job many times. Job templates also encourage the reuse of Ansible Playbook content and collaboration between teams. For more information, see [Job Templates](#) in the Using automation execution guide.

Platform administrators and automation developers have the permissions to create job templates. As an automation operator you can launch job templates and view their details.

4.8.1. Launching a job template

Ansible Automation Platform offers push-button deployment of Ansible playbooks. You can configure a template to store all the parameters that you would normally pass to the Ansible playbook on the command line. In addition to the playbooks, the template passes the inventory, credentials, extra variables, and all options and settings that you can specify on the command line.

Procedure

1. From the navigation panel, select **Automation Execution → Templates**.
2. Select a template to view its details. A default job template is created during your initial setup to help you get started, but you can also create your own.
3. From the **Templates** page, click the launch icon to run your job template.

The **Templates** list view shows job templates that are currently available. The default view is collapsed (Compact), showing the template name, template type, and the timestamp of the last job that ran using that template. You can click the arrow icon next to each entry to expand and view more information. This list is sorted alphabetically by name, but you can sort by other criteria, or search by various template fields and attributes.

From this screen you can launch, edit, and copy a job template.

For more information about templates see the [Job templates](#) and [Workflow job templates](#) sections of the Using automation execution.

4.9. ABOUT INVENTORIES

An inventory is a file listing the collection of hosts managed by Ansible Automation Platform. Organizations are assigned to inventories, while permissions to launch playbooks against inventories are controlled at the user or team level.

Platform administrators and automation developers have the permissions to create inventories. As an automation operator you can view inventories and their details.

4.9.1. Executing an inventory

Procedure

1. From the navigation panel, select **Automation Execution → Infrastructure → Inventories**. The **Inventories** window displays a list of inventories that are currently available, along with the following information:
 - **Name:** The inventory name.
 - **Status:** The statuses are:
 - **Success:** The inventory sync completed successfully.
 - **Disabled:** No inventory source added to the inventory.
 - **Error:** The inventory source completed with error.
 - **Type:** Identifies whether the inventory is a standard inventory, a smart inventory, or a constructed inventory.

- **Organization:** The organization to which the inventory belongs.
2. Select an inventory name to display the **Details** page for the inventory, including the inventory's groups and hosts.

For more information about inventories, see the [Inventories](#) section of the Using automation execution guide.

4.10. AUTOMATION EXECUTION JOBS

A job is an instance of Ansible Automation Platform launching an Ansible Playbook against an inventory of hosts.

4.10.1. Reviewing a job status

The **Jobs** list view displays a list of jobs and their statuses, shown as completed successfully, failed, or as an active (running) job.

Procedure

1. From the navigation panel, select **Automation Execution → Jobs**.
The default view is collapsed (Compact) with the job name, status, job type, start, and finish times. You can click the arrow icon to expand and see more information. You can sort this list by various criteria, and perform a search to filter the jobs of interest.
2. From this screen, you can complete the following tasks:
 - View a job's details and standard output.
 - Relaunch jobs.
 - Remove selected jobs.

The relaunch operation only applies to relaunches of playbook runs and does not apply to project or inventory updates, system jobs, or workflow jobs.

4.10.2. Reviewing job output

When you relaunch a job, the jobs **Output** view is displayed.

Procedure

1. From the navigation panel, select **Automation Execution → Jobs**.
2. Select a job. This takes you to the **Output** view for that job, where you can filter job output by these criteria:
 - The **Search output** option allows you to search by keyword.
 - The **Event** option enables you to filter by the events of interest, such as errors, host failures, host retries, and items skipped. You can include as many events in the filter as necessary.

