



Red Hat Ansible Automation Platform 2.4

Automation Content Navigator Creator Guide

Develop content that is compatible with Ansible Automation Platform

Red Hat Ansible Automation Platform 2.4 Automation Content Navigator Creator Guide

Develop content that is compatible 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 use automation content navigator to develop Ansible playbooks, collections, and roles that are compatible with Ansible Automation Platform.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE	3
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	4
CHAPTER 1. INTRODUCTION TO AUTOMATION CONTENT NAVIGATOR	5
1.1. USES FOR AUTOMATION CONTENT NAVIGATOR	5
1.2. AUTOMATION CONTENT NAVIGATOR MODES	5
1.2.1. stdout mode	5
1.2.2. Text-based user interface mode	6
1.3. AUTOMATION CONTENT NAVIGATOR COMMANDS	6
1.4. RELATIONSHIP BETWEEN ANSIBLE AND AUTOMATION CONTENT NAVIGATOR COMMANDS	7
CHAPTER 2. INSTALLING AUTOMATION CONTENT NAVIGATOR ON RHEL	8
2.1. INSTALLING AUTOMATION CONTENT NAVIGATOR ON RHEL FROM AN RPM	8
CHAPTER 3. REVIEWING AUTOMATION EXECUTION ENVIRONMENTS WITH AUTOMATION CONTENT NAVIGATOR	10
3.1. REVIEWING AUTOMATION EXECUTION ENVIRONMENTS FROM AUTOMATION CONTENT NAVIGATOR	10
CHAPTER 4. REVIEWING INVENTORIES WITH AUTOMATION CONTENT NAVIGATOR	11
4.1. REVIEWING INVENTORY FROM AUTOMATION CONTENT NAVIGATOR	11
CHAPTER 5. BROWSING COLLECTIONS WITH AUTOMATION CONTENT NAVIGATOR	13
5.1. AUTOMATION CONTENT NAVIGATOR COLLECTIONS DISPLAY	13
5.2. BROWSING COLLECTIONS FROM AUTOMATION CONTENT NAVIGATOR	13
5.3. REVIEW DOCUMENTATION FROM AUTOMATION CONTENT NAVIGATOR	15
CHAPTER 6. RUNNING ANSIBLE PLAYBOOKS WITH AUTOMATION CONTENT NAVIGATOR	18
6.1. EXECUTING A PLAYBOOK FROM AUTOMATION CONTENT NAVIGATOR	18
6.2. REVIEWING PLAYBOOK RESULTS WITH AN AUTOMATION CONTENT NAVIGATOR ARTIFACT FILE	19
CHAPTER 7. REVIEWING YOUR ANSIBLE CONFIGURATION WITH AUTOMATION CONTENT NAVIGATOR ..	21
7.1. REVIEWING YOUR ANSIBLE CONFIGURATION FROM AUTOMATION CONTENT NAVIGATOR	21
CHAPTER 8. AUTOMATION CONTENT NAVIGATOR CONFIGURATION SETTINGS	23
8.1. CREATING AN AUTOMATION CONTENT NAVIGATOR SETTINGS FILE	23
8.2. AUTOMATION CONTENT NAVIGATOR GENERAL SETTINGS	24
8.3. AUTOMATION CONTENT NAVIGATOR CONFIG SUBCOMMAND SETTINGS	32
8.4. AUTOMATION CONTENT NAVIGATOR DOC SUBCOMMAND SETTINGS	33
8.5. AUTOMATION CONTENT NAVIGATOR INVENTORY SUBCOMMAND SETTINGS	34
8.6. AUTOMATION CONTENT NAVIGATOR REPLAY SUBCOMMAND SETTINGS	35
8.7. AUTOMATION CONTENT NAVIGATOR RUN SUBCOMMAND SETTINGS	36
CHAPTER 9. TROUBLESHOOTING ANSIBLE CONTENT WITH AUTOMATION CONTENT NAVIGATOR ..	39
9.1. REVIEWING PLAYBOOK RESULTS WITH AN AUTOMATION CONTENT NAVIGATOR ARTIFACT FILE	39
9.2. FREQUENTLY ASKED QUESTIONS ABOUT AUTOMATION CONTENT NAVIGATOR	39

MAKING OPEN SOURCE MORE INCLUSIVE

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

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

If you have a suggestion to improve this documentation, or find an error, please contact technical support at <https://access.redhat.com> to create an issue on the Ansible Automation Platform Jira project using the **docs-product** component.

CHAPTER 1. INTRODUCTION TO AUTOMATION CONTENT NAVIGATOR

As a content creator, you can use automation content navigator to develop Ansible playbooks, collections, and roles that are compatible with the Red Hat Ansible Automation Platform. You can use automation content navigator in the following environments, with seamless and predictable results across them all:

- Local development machines
- Automation execution environments

Automation content navigator also produces an artifact file you can use to help you develop your playbooks and troubleshoot problem areas.

1.1. USES FOR AUTOMATION CONTENT NAVIGATOR

Automation content navigator is a command line, content-creator-focused tool with a text-based user interface. You can use automation content navigator to:

- Launch and watch jobs and playbooks.
- Share stored, completed playbook and job run artifacts in JSON format.
- Browse and introspect automation execution environments.
- Browse your file-based inventory.
- Render Ansible module documentation and extract examples you can use in your playbooks.
- View a detailed command output on the user interface.

1.2. AUTOMATION CONTENT NAVIGATOR MODES

Automation content navigator operates in two modes:

stdout mode

Accepts most of the existing Ansible commands and extensions at the command line.

text-based user interface mode

Provides an interactive, text-based interface to the Ansible commands. Use this mode to evaluate content, run playbooks, and troubleshoot playbooks after they run using artifact files.

1.2.1. stdout mode

Use the **-m stdout** subcommand with automation content navigator to use the familiar Ansible commands, such as **ansible-playbook** within automation execution environments or on your local development environment. You can use commands you are familiar with for quick tasks.

Automation content navigator also provides extensive help in this mode:

--help

Accessible from **ansible-navigator** command or from any subcommand, such as **ansible-navigator config --help**.

subcommand help

Accessible from the subcommand, for example **ansible-navigator config --help-config**. This help displays the details of all the parameters supported from the related Ansible command.

1.2.2. Text-based user interface mode

The text-based user interface mode provides enhanced interaction with automation execution environments, collections, playbooks, and inventory. This mode is compatible with integrated development environments (IDE), such as Visual Studio Code.

```

0 ## Welcome
1 -----
2
3 Some things you can try from here:
4 - `:collections`      Explore available collections
5 - `:config`          Explore the current ansible configuration
6 - `:doc <plugin>`    Review documentation for a module or plugin
7 - `:help`            Show the main help page
8 - `:images`         Explore execution environment images
9 - `:inventory -i <inventory>` Explore an inventory
10 - `:log`            Review the application log
11 - `:open`           Open current page in the editor
12 - `:replay`        Explore a previous run using a playbook artifact
13 - `:run <playbook> -i <inventory>` Run a playbook in interactive mode
14 - `:quit`          Quit the application
15
16 happy automating,
17
18 -winston

```

`^f/PgUp` page up `^b/PgDn` page down `↑` scroll `esc` back `:help` help

This mode includes a number of helpful user interface options:

colon commands

You can access all the automation content navigator commands with a colon, such as **:run** or **:collections**

navigating the text-based interface

The screen shows how to page up or down, scroll, escape to a prior screen or access **:help**.

output by line number

You can access any line number in the displayed output by preceding it with a colon, for example **:12**.

color-coded output

With colors enabled, automation content navigator displays items, such as deprecated modules, in red.

pagination and scrolling

You can page up or down, scroll, or escape by using the options displayed at the bottom of each automation content navigator screen.

You cannot switch between modes after automation content navigator is running.

This document uses the text-based user interface mode for most procedures.

1.3. AUTOMATION CONTENT NAVIGATOR COMMANDS

The automation content navigator commands run familiar Ansible CLI commands in **-m stdout** mode. You can use all the subcommands and options from the related Ansible CLI command. Use **ansible-navigator --help** for details.

Table 1.1. Automation content navigator commands

Command	Description	CLI example
collections	Explore available collections	ansible-navigator collections --help
config	Explore the current Ansible configuration	ansible-navigator config --help
doc	Review documentation for a module or plugin	ansible-navigator doc --help
images	Explore execution environment images	ansible-navigator images --help
inventory	Explore an inventory	ansible-navigator inventory -help
replay	Explore a previous run using a playbook artifact	ansible-navigator replay --help
run	Run a playbook	ansible-navigator run --help
welcome	Start at the welcome page	ansible-navigator welcome --help

1.4. RELATIONSHIP BETWEEN ANSIBLE AND AUTOMATION CONTENT NAVIGATOR COMMANDS

The automation content navigator commands run familiar Ansible CLI commands in **-m stdout** mode. You can use all the subcommands and options available in the related Ansible CLI command. Use **ansible-navigator --help** for details.

Table 1.2. Comparison of automation content navigator and Ansible CLI commands

automation content navigator command	Ansible CLI command
ansible-navigator collections	ansible-galaxy collection
ansible-navigator config	ansible-config
ansible-navigator doc	ansible-doc
ansible-navigator inventory	ansible-inventory
ansible-navigator run	ansible-playbook

CHAPTER 2. INSTALLING AUTOMATION CONTENT NAVIGATOR ON RHEL

As a content creator, you can install automation content navigator on Red Hat Enterprise Linux (RHEL) 8.6 or later.

2.1. INSTALLING AUTOMATION CONTENT NAVIGATOR ON RHEL FROM AN RPM

You can install automation content navigator on Red Hat Enterprise Linux (RHEL) from an RPM.

Prerequisites

- You have installed RHEL 8.6 or later.
- You registered your system with Red Hat Subscription Manager.



NOTE

Ensure that you only install the navigator matching your current Red Hat Ansible Automation Platform environment.

Procedure

1. Attach the Red Hat Ansible Automation Platform SKU:

```
$ subscription-manager attach --pool=<sku-pool-id>
```

2. Install automation content navigator with the following command:
v.2.4 for RHEL 8 for x86_64

```
$ sudo dnf install --enablerepo=ansible-automation-platform-2.4-for-rhel-8-x86_64-rpms  
ansible-navigator
```

v.2.4 for RHEL 9 for x86-64

```
$ sudo dnf install --enablerepo=ansible-automation-platform-2.4-for-rhel-9-x86_64-rpms  
ansible-navigator
```

Verification

- Verify your automation content navigator installation:

```
$ ansible-navigator --help
```

The following example demonstrates a successful installation:

```
$ ansible-navigator --help
usage: ansible-navigator [-h] [--version] [--cdcp COLLECTION_DOC_CACHE_PATH] [--ce CONTAINER_ENGINE] [--dc DISPLAY_COLOR] [--ecmd EDITOR_COMMAND]
                        [--econ EDITOR_CONSOLE] [--ee EXECUTION_ENVIRONMENT] [--eei EXECUTION_ENVIRONMENT_IMAGE]
                        [--eev EXECUTION_ENVIRONMENT_VOLUME_MOUNTS [EXECUTION_ENVIRONMENT_VOLUME_MOUNTS ...]] [--la LOG_APPEND] [--lf LOG_FILE]
                        [--ll LOG_LEVEL] [-m MODE] [--osc4 OSC4] [--penv PASS_ENVIRONMENT_VARIABLE [PASS_ENVIRONMENT_VARIABLE ...]]
                        [--pp PULL_POLICY] [--senv SET_ENVIRONMENT_VARIABLE [SET_ENVIRONMENT_VARIABLE ...]]
                        {subcommand} --help ...

optional arguments:
  -h, --help            show this help message and exit
  --version             show program's version number and exit

<... output truncated ...>

Subcommands:
{subcommand} --help
collections            Explore available collections
config                 Explore the current ansible configuration
doc                    Review documentation for a module or plugin
images                 Explore execution environment images
inventory              Explore an inventory
replay                 Explore a previous run using a playbook artifact
run                    Run a playbook
welcome                Start at the welcome page
```

CHAPTER 3. REVIEWING AUTOMATION EXECUTION ENVIRONMENTS WITH AUTOMATION CONTENT NAVIGATOR

As a content developer, you can review your automation execution environment with automation content navigator and display the packages and collections included in the automation execution environments. Automation content navigator runs a playbook to extract and display the results.

3.1. REVIEWING AUTOMATION EXECUTION ENVIRONMENTS FROM AUTOMATION CONTENT NAVIGATOR

You can review your automation execution environments with the automation content navigator text-based user interface.

Prerequisites

- Automation execution environments

Procedure

- Review the automation execution environments included in your automation content navigator configuration.

```
$ ansible-navigator images
```

NAME	TAG	EXECUTION ENVIRONMENT	CREATED	SIZE
0 ansible-automation-platform-20-ee-minimal-rhel8	latest	True	4 weeks ago	411 MB
1 ansible-automation-platform-20-ee-supported-rhel8 (primary)	latest	True	45 hours ago	923 MB
2 ansible-runner	devel	True	4 weeks ago	652 MB
3 ccutil	amazing	False	16 months ago	1.67 GB

- Type the number of the automation execution environment you want to delve into for more details.

ANSIBLE - AUTOMATION - PLATFORM - 20 - EE - SUPPORTED - RHEL8 : LATEST (PRIMARY)	DESCRIPTION
0 Image information	Information collected from image inspection
1 General information	OS and python version information
2 Ansible version and collections	Information about ansible and ansible collections
3 Python packages	Information about python and python packages
4 Operating system packages	Information about operating system packages
5 Everything	All image information

You can review the packages and versions of each installed automation execution environment and the Ansible version any included collections.

- Optional: pass in the automation execution environment that you want to use. This becomes the primary and is the automation execution environment that automation content navigator uses.

```
$ ansible-navigator images --eei registry.example.com/example-enterprise-ee:latest
```

Verification

- Review the automation execution environment output.

ANSIBLE - AUTOMATION - PLATFORM - 20 - EE - SUPPORTED - RHEL8 : LATEST (PRIMARY)	DESCRIPTION
0 Image information	Information collected from image inspection
1 General information	OS and python version information
2 Ansible version and collections	Information about ansible and ansible collections
3 Python packages	Information about python and python packages
4 Operating system packages	Information about operating system packages
5 Everything	All image information

CHAPTER 4. REVIEWING INVENTORIES WITH AUTOMATION CONTENT NAVIGATOR

As a content creator, you can review your Ansible inventory with automation content navigator and interactively delve into the groups and hosts.

4.1. REVIEWING INVENTORY FROM AUTOMATION CONTENT NAVIGATOR

You can review Ansible inventories with the automation content navigator text-based user interface in interactive mode and delve into groups and hosts for more details.

Prerequisites

- A valid inventory file or an inventory plugin.

Procedure

1. Start automation content navigator.

```
$ ansible-navigator
```

Optional: type **ansible-navigator inventory -i simple_inventory.yml** from the command line to view the inventory.

2. Review the inventory.

```
:inventory -i simple_inventory.yml

  TITLE      DESCRIPTION
0| Browse groups  Explore each inventory group and group members members
1| Browse hosts   Explore the inventory with a list of all hosts
```

3. Type **0** to brows the groups.

```
  NAME      TAXONOMY      TYPE
0| general   all           group
1| nodes     all           group
2| ungrouped all           group
```

The **TAXONOMY** field details the hierarchy of groups the selected group or node belongs to.

4. Type the number corresponding to the group you want to delve into.

```
  NAME      TAXONOMY      TYPE
0| node-0    all▶nodes     host
1| node-1    all▶nodes     host
2| node-2    all▶nodes     host
```

5. Type the number corresponding to the host you want to delve into, or type **:<number>** for numbers greater than 9.

```
[node-1]
0| ---
1| ansible_host: node-1.example.com
2| inventory_hostname: node-1
```

Verification

- Review the inventory output.

```
TITLE      DESCRIPTION
0| Browse groups  Explore each inventory group and group members members
1| Browse hosts   Explore the inventory with a list of all hosts
```

Additional resources

- [ansible-inventory](#).
- [How to build your inventory](#) .

CHAPTER 5. BROWSING COLLECTIONS WITH AUTOMATION CONTENT NAVIGATOR

As a content creator, you can browse your Ansible collections with automation content navigator and interactively delve into each collection developed locally or within Automation execution environments.

5.1. AUTOMATION CONTENT NAVIGATOR COLLECTIONS DISPLAY

Automation content navigator displays information about your collections with the following details for each collection:

SHADOWED

Indicates that an additional copy of the collection is higher in the search order, and playbooks prefer that collection.

TYPE

Shows if the collection is contained within an automation execution environment or volume mounted on onto the automation execution environment as a **bind_mount**.

PATH

Reflects the collections location within the automation execution environment or local file system based on the collection TYPE field.

	NAME	VERSION	SHADOWED	TYPE	PATH
0	amazon.aws	1.5.0	False	contained	/usr/share/ansible/collections/ansible_collections/amazon/aws/
1	ansible.posix	1.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/posix/
2	ansible.utils	2.2.0	False	bind_mount	/home/samccann/aap/collections/ansible_collections/ansible/utils/
3	ansible.windows	1.6.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/windows/
4	awx.awx	19.2.2	False	bind_mount	/home/samccann/aap/collections/ansible_collections/awx/awx/
5	awx.awx	19.2.0	True	contained	/usr/share/ansible/collections/ansible_collections/awx/awx/
6	azure.azcollection	1.7.0	False	contained	/usr/share/ansible/collections/ansible_collections/azure/azcollection/

5.2. BROWSING COLLECTIONS FROM AUTOMATION CONTENT NAVIGATOR

You can browse Ansible collections with the automation content navigator text-based user interface in interactive mode and delve into each collection. automation content navigator shows collections within the current project directory and those available in the automation execution environments

Prerequisites

- A locally accessible collection or installed automation execution environments.

Procedure

1. Start automation content navigator

```
$ ansible-navigator
```

2. Browse the collection. Alternately, you can type **ansible-navigator collections** to directly browse the collections.

```
$ :collections
```

	NAME	VERSION	SHADOWED	TYPE	PATH
0	amazon.aws	1.4.1	False	contained	/usr/share/ansible/collections/ansible_collections/amazon/aws/
1	ansible.netcommon	2.1.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/netcommon/
2	ansible.posix	1.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/posix/
3	ansible.tower	3.8.3	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/tower/
4	ansible.utils	2.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/utils/
5	ansible.windows	1.5.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/windows/
6	arista.eos	2.1.2	False	contained	/usr/share/ansible/collections/ansible_collections/arista/eos/

- Type the number of the collection you want to explore.

```
:4
```

	ANSIBLE.UTILS	TYPE	ADDED	DEPRECATED	DESCRIPTION
0	cli_parse	module	1.0.0	False	Parse cli output or text using a variety of parsers
1	fact_diff	module	1.0.0	False	Find the difference between currently set facts
2	from_xml	filter	2.0.2	False	Convert given XML string to native python dictionary.
3	get_path	lookup	1.0.0	False	Retrieve the value in a variable using a path
4	get_path	filter	1.0.0	False	Retrieve the value in a variable using a path
5	in_any_network	test	2.2.0	False	Test if an IP or network falls in any network
6	in_network	test	2.2.0	False	Test if IP address falls in the network

- Type the number corresponding to the module you want to delve into.

```
ANSIBLE.UTILS.IP_ADDRESS: Test if something in an IP address
0 | ---
1 | additional_information: {}
2 | collection_info:
3 |   authors:
4 |     - Ansible Community
5 |   dependencies: {}
6 |   description: Ansible Collection with utilities to ease the management, manipulation,
7 |     and validation of data within a playbook
8 |   documentation: null
9 |   homepage: null
10 |   issues: null
11 |   license: []
12 |   license_file: LICENSE
13 |   name: ansible.utils
14 |   namespace: ansible
15 |   path:/usr/share/ansible/collections/ansible_collections/ansible/utils/
16 |   readme: README.md
<... output truncated...>
```

- Optional: jump to the documentation examples for this module.

```
:{{ examples }}
0 |
1 |
2 | ##### Simple examples
3 |
4 | - name: Check if 10.1.1.1 is a valid IP address
5 |   ansible.builtin.set_fact:
6 |     data: "{{ '10.1.1.1' is ansible.utils.ip_address }}"
7 |
8 | # TASK [Check if 10.1.1.1 is a valid IP address] *****
9 | # ok: [localhost] => {
10 | #   "ansible_facts": {
11 | #     "data": true
12 | #   },
```

```

13 | # "changed": false
14 | # }
15 |

```

- Optional: open the example in your editor to copy it into a playbook.

```

:open

```

```

EXPLORE... tmpvsqroz08.yaml x
> .ansible
> vagrant-test
> .gitignore
> ansible-navigator.log
! ansible-navigator.yml
! ansible.cfg
! quicklab_inventory.yml
{} quicklab_playbook-artifa... U
{} quicklab_playbook-artifa... U
{} quicklab_playbook-artifa... U
{} quicklab_playbook-artifa... U
! quicklab_playbook.yml
@ README.md
{} simple_playbook-artifact... U
{} simple_playbook-artifact... U
{} simple_playbook-artifact... U
{} simple_playbook-artifact... U
{} simple_playbook-artifact... U
{} simple_playbook-artifact... U

51 version_added_collection: ansible.utils
52 examples: |-
53     ##### Simple examples
54
55     - name: Check if 10.1.1.1 is in 10.0.0.0/8
56       ansible.builtin.set_fact:
57         data: "{{ '10.1.1.1' is ansible.utils.in_network '10.0.0.0/8' }}"
58
59     # TASK [Check if 10.1.1.1 is in 10.0.0.0/8] *****
60     # ok: [localhost] => {
61     #   "ansible_facts": {
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
1: python3
ANSIBLE_UTILS.IN_NETWORK: Test if IP address falls in the network
0 ---
1 additional_information: {}
2 collection_info:
3   authors:
4     - Ansible Community
5   dependencies: {}
6   description: Ansible Collection with utilities to ease the management, manipulation,
7     and validation of data within a playbook
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Verification

- Browse the collection list.

	NAME	VERSION	SHADOWED	TYPE	PATH
0	amazon.aws	1.4.1	False	contained	/usr/share/ansible/collections/ansible_collections/amazon/aws/
1	ansible.netcommon	2.1.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/netcommon/
2	ansible.posix	1.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/posix/
3	ansible.tower	3.8.3	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/tower/
4	ansible.utils	2.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/utils/
5	ansible.windows	1.5.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/windows/
6	arista.eos	2.1.2	False	contained	/usr/share/ansible/collections/ansible_collections/arista/eos/

Additional resources

- [ansible-collection](#).
- [Introduction to Ansible inventories](#).

5.3. REVIEW DOCUMENTATION FROM AUTOMATION CONTENT NAVIGATOR

You can review Ansible documentation for collections and plugins with the automation content navigator text-based user interface in interactive mode. automation content navigator shows collections within the current project directory and those available in the automation execution environments

Prerequisites

- A locally accessible collection or installed automation execution environments.

Procedure

1. Start automation content navigator

```
$ ansible-navigator
```

2. Review the module you are interested in. Alternately, you can type **ansible-navigator doc** to access the documentation.

```
:doc ansible.utils.ip_address
```

```
ANSIBLE.UTILS.IP_ADDRESS: Test if something in an IP address
0 | ---
1 | additional_information: {}
2 | collection_info:
3 |   authors:
4 |     - Ansible Community
5 |   dependencies: {}
6 |   description: Ansible Collection with utilities to ease the management, manipulation,
7 |     and validation of data within a playbook
8 |   documentation: null
9 |   homepage: null
10 |  issues: null
11 |  license: []
12 |  license_file: LICENSE
13 |  name: ansible.utils
14 |  namespace: ansible
15 |  path: /usr/share/ansible/collections/ansible_collections/ansible/utils/
16 |  readme: README.md
<... output truncated...>
```

3. Jump to the documentation examples for this module.

```
:{{ examples }}
0 |
1 |
2 | ##### Simple examples
3 |
4 | - name: Check if 10.1.1.1 is a valid IP address
5 |   ansible.builtin.set_fact:
6 |     data: "{{ '10.1.1.1' is ansible.utils.ip_address }}"
7 |
8 | # TASK [Check if 10.1.1.1 is a valid IP address] *****
9 | # ok: [localhost] => {
10 | #   "ansible_facts": {
11 | #     "data": true
12 | #   },
13 | #   "changed": false
14 | # }
15 |
```

4. Optional: open the example in your editor to copy it into a playbook.

```
:open
```

The screenshot shows an IDE window with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with files like `.ansible`, `vagrant-test`, `.gitignore`, `ansible-navigator.log`, `ansible-navigator.yml`, `ansible.cfg`, `quicklab_inventory.yml`, `quicklab_playbook-artifa...`, `README.md`, and `simple_playbook-artifact...`. The code editor shows the content of `tmpvsqroz08.yaml`:

```

51 version_added_collection: ansible.utils
52 examples: |-
53     ### Simple examples
54
55     - name: Check if 10.1.1.1 is in 10.0.0.0/8
56       ansible.builtin.set_fact:
57         data: "{{ '10.1.1.1' is ansible.utils.in_network '10.0.0.0/8' }}"
58
59     # TASK [Check if 10.1.1.1 is in 10.0.0.0/8] *****
60     # ok: [localhost] => {
61     #   "ansible_facts": {

```

Below the code editor, there is a terminal window showing the output of the Ansible command:

```

ANSIBLE_UTILS_IN_NETWORK: Test if IP address falls in the network
0 ---
1 additional_information: {}
2 collection_info:
3   authors:
4     - Ansible Community
5   dependencies: {}
6   description: Ansible Collection with utilities to ease the management, manipulation,
7     and validation of data within a playbook

```

See [Automation content navigator general settings](#) for details on how to set up your editor.

Additional resources

- [Collection index](#)
- [Using Ansible collections](#)
- [Building Ansible inventories](#)

CHAPTER 6. RUNNING ANSIBLE PLAYBOOKS WITH AUTOMATION CONTENT NAVIGATOR

As a content creator, you can execute your Ansible playbooks with automation content navigator and interactively delve into the results of each play and task to verify or troubleshoot the playbook. You can also execute your Ansible playbooks inside an execution environment and without an execution environment to compare and troubleshoot any problems.

6.1. EXECUTING A PLAYBOOK FROM AUTOMATION CONTENT NAVIGATOR

You can run Ansible playbooks with the automation content navigator text-based user interface to follow the execution of the tasks and delve into the results of each task.

Prerequisites

- A playbook.
- A valid inventory file if not using **localhost** or an inventory plugin.

Procedure

1. Start automation content navigator

```
$ ansible-navigator
```

2. Run the playbook.

```
$ :run
```

3. Optional: type **ansible-navigator run simple-playbook.yml -i inventory.yml** to run the playbook.
4. Verify or add the inventory and any other command line parameters.

```
INVENTORY OR PLAYBOOK NOT FOUND, PLEASE CONFIRM THE FOLLOWING
```

```
Path to playbook: /home/ansible-navigator_demo/simple_playbook.yml
Inventory source: /home/ansible-navigator-demo/inventory.yml
Additional command line parameters: Please provide a value (optional)
```

Submit Cancel

5. Tab to **Submit** and hit Enter. You should see the tasks executing.

PLAY NAME	OK	CHANGED	UNREACHABLE	FAILED	SKIPPED	IGNORED	IN PROGRESS	TASK COUNT	PROGRESS
all	6	0	0	6	0	0	0	12	COMPLETE

6. Type the number next to a play to step into the play results, or type **:<number>** for numbers above 9.

	RESULT	HOST	NUMBER	CHANGED	TASK	TASK ACTION	DURATION
3	OK	node-0	3	False	Gathering Facts	gather_facts	1s
4	OK	node-1	4	False	Gathering Facts	gather_facts	1s
5	OK	node-2	5	False	Gathering Facts	gather_facts	1s
6	FAILED	main-0	6	False	Gather the package facts	ansible.builtin.package_facts	1s
7	FAILED	infra-0	7	False	Gather the package facts	ansible.builtin.package_facts	1s
8	FAILED	lb-0	8	False	Gather the package facts	ansible.builtin.package_facts	1s
9	FAILED	node-0	9	False	Gather the package facts	ansible.builtin.package_facts	1s
10	FAILED	node-1	10	False	Gather the package facts	ansible.builtin.package_facts	1s
11	FAILED	node-2	11	False	Gather the package facts	ansible.builtin.package_facts	0s

Notice failed tasks show up in red if you have colors enabled for automation content navigator.

7. Type the number next to a task to review the task results, or type **:<number>** for numbers above 9.

```
PLAY [all:6] *****
TASK [Gather the package facts] *****
FAILED: [main-0] Could not detect a supported package manager from the following list: ['apt', 'apk', 'rpm', 'portage', 'pkg']
0 | ---
1 | duration: 1.339719
2 | end: '2021-06-10T18:52:32.968770'
3 | event_loop: null
4 | host: main-0
5 | ignore_errors: null
6 | play: all
```

8. Optional: type **:doc** bring up the documentation for the module or plugin used in the task to aid in troubleshooting.

```
ANSIBLE.BUILTIN.PACKAGE_FACTS (MODULE)
0 | ---
1 | doc:
2 |   author:
3 |     - Matthew Jones (@matburt)
4 |     - Brian Coca (@bcoca)
5 |     - Adam Miller (@maxamillion)
6 |   collection: ansible.builtin
7 |   description:
8 |     - Return information about installed packages as facts.
<... output omitted ...>
11 | module: package_facts
12 | notes:
13 |   - Supports C(check_mode).
14 | options:
15 |   manager:
16 |     choices:
17 |       - auto
18 |       - rpm
19 |       - apt
20 |       - portage
21 |       - pkg
22 |       - pacman
<... output truncated ...>
```

Additional resources

- [ansible-playbook](#)
- [Ansible playbooks](#)

6.2. REVIEWING PLAYBOOK RESULTS WITH AN AUTOMATION CONTENT NAVIGATOR ARTIFACT FILE

Automation content navigator saves the results of the playbook run in a JSON artifact file. You can use this file to share the playbook results with someone else, save it for security or compliance reasons, or review and troubleshoot later. You only need the artifact file to review the playbook run. You do not need access to the playbook itself or inventory access.

Prerequisites

- A automation content navigator artifact JSON file from a playbook run.

Procedure

- Start automation content navigator with the artifact file.

```
$ ansible-navigator replay simple_playbook_artifact.json
```

1. Review the playbook results that match when the playbook ran.

PLAY NAME	OK	CHANGED	UNREACHABLE	FAILED	SKIPPED	IGNORED	IN PROGRESS	TASK COUNT	PROGRESS
all	12	0	0	0	25	0	0	37	COMPLETE

You can now type the number next to the plays and tasks to step into each to review the results, as you would after executing the playbook.

Additional resources

- [ansible-playbook](#)
- [Ansible playbooks](#)

CHAPTER 7. REVIEWING YOUR ANSIBLE CONFIGURATION WITH AUTOMATION CONTENT NAVIGATOR

As a content creator, you can review your Ansible configuration with automation content navigator and interactively delve into settings.

7.1. REVIEWING YOUR ANSIBLE CONFIGURATION FROM AUTOMATION CONTENT NAVIGATOR

You can review your Ansible configuration with the automation content navigator text-based user interface in interactive mode and delve into the settings. Automation content navigator pulls in the results from an accessible Ansible configuration file, or returns the defaults if no configuration file is present.

Prerequisites

- You have authenticated to the Red Hat registry if you need to access additional automation execution environments. See [Red Hat Container Registry Authentication](#) for details.

Procedure

- Start automation content navigator

```
$ ansible-navigator
```

Optional: type **ansible-navigator config** from the command line to access the Ansible configuration settings.

- Review the Ansible configuration.

```
:config
```

OPTION	DEFAULT	SOURCE	VIA	CURRENT VALUE
0 ACTION_WARNINGS	True	default	default	True
1 AGNOSTIC_BECOME_PROMPT	True	default	default	True
2 ALLOW_WORLD_READABLE_TMPFI	True	default	default	False
3 ANSIBLE_CONNECTION_PATH	True	default	default	None
4 ANSIBLE_COW_ACCEPTLIST	False	/home/samccann/ansible-nav/home/samccann/ansible-nav['bud-frogs', 'bunny',		None
5 ANSIBLE_COW_PATH	True	default	default	default
6 ANSIBLE_COW_SELECTION	True	default	default	False
7 ANSIBLE_FORCE_COLOR	True	default	default	False

Some values reflect settings from within the automation execution environments needed for the automation execution environments to function. These display as non-default settings you cannot set in your Ansible configuration file.

- Type the number corresponding to the setting you want to delve into, or type **:<number>** for numbers greater than 9.

```
ANSIBLE COW ACCEPTLIST (current: ['bud-frogs', 'bunny', 'cheese']) (default:
0 | ---
1 | current:
2 | - bud-frogs
3 | - bunny
4 | - cheese
5 | default:
```

```

6 | - bud-frogs
7 | - bunny
8 | - cheese
9 | - daemon

```

The output shows the current **setting** as well as the **default**. Note the **source** in this example is **env** since the setting comes from the automation execution environments.

Verification

- Review the configuration output.

	OPTION	DEFAULT	SOURCE	VIA	CURRENT VALUE
0	ACTION_WARNINGS	True	default	default	True
1	AGNOSTIC_BECOME_PROMPT	True	default	default	True
2	ALLOW_WORLD_READABLE_TMPFI	True	default	default	False
3	ANSIBLE_CONNECTION_PATH	True	default	default	None
4	ANSIBLE_COW_ACCEPTLIST	False	/home/samccann/ansible-nav/home/samccann/ansible-nav['bud-frogs', 'bunny',		
5	ANSIBLE_COW_PATH	True	default	default	None
6	ANSIBLE_COW_SELECTION	True	default	default	default
7	ANSIBLE_FORCE_COLOR	True	default	default	False

Additional resources

- [ansible-config](#).
- [Introduction to Ansible configuration](#).

CHAPTER 8. AUTOMATION CONTENT NAVIGATOR CONFIGURATION SETTINGS

As a content creator, you can configure automation content navigator to suit your development environment.

8.1. CREATING AN AUTOMATION CONTENT NAVIGATOR SETTINGS FILE

You can alter the default automation content navigator settings through:

- The command line
- Within a settings file
- As an environment variable

Automation content navigator checks for a settings file in the following order and uses the first match:

- **ANSIBLE_NAVIGATOR_CONFIG** - The settings file path environment variable if set.
- **./ansible-navigator.<ext>** - The settings file within the current project directory, with no dot in the file name.
- **~/ansible-navigator.<ext>** - Your home directory, with a dot in the file name.

Consider the following when you create an automation content navigator settings file:

- The settings file can be in **JSON** or **YAML** format.
- For settings in **JSON** format, the extension must be **.json**.
- For settings in **YAML** format, the extension must be **.yaml** or **.yml**.
- The project and home directories can only contain one settings file each.
- If automation content navigator finds more than one settings file in either directory, it results in an error.

You can copy the example settings file below into one of those paths to start your **ansible-navigator** settings file.

```
---
ansible-navigator:
# ansible:
# config: /tmp/ansible.cfg
# cmdline: "--forks 15"
# inventories:
# - /tmp/test_inventory.yml
# playbook: /tmp/test_playbook.yml
# ansible-runner:
# artifact-dir: /tmp/test1
# rotate-artifacts-count: 10
# timeout: 300
# app: run
```

```

# collection-doc-cache-path: /tmp/cache.db
# color:
# enable: False
# osc4: False
# editor:
# command: vim_from_setting
# console: False
# documentation:
# plugin:
# name: shell
# type: become
# execution-environment:
# container-engine: podman
# enabled: False
# environment-variables:
# pass:
# - ONE
# - TWO
# - THREE
# set:
# KEY1: VALUE1
# KEY2: VALUE2
# KEY3: VALUE3
# image: test_image:latest
# pull-policy: never
# volume-mounts:
# - src: "/test1"
# dest: "/test1"
# label: "Z"
# help-config: True
# help-doc: True
# help-inventory: True
# help-playbook: False
# inventory-columns:
# - ansible_network_os
# - ansible_network_cli_ssh_type
# - ansible_connection
logging:
# append: False
level: critical
# file: /tmp/log.txt
# mode: stdout
# playbook-artifact:
# enable: True
# replay: /tmp/test_artifact.json
# save-as: /tmp/test_artifact.json

```

8.2. AUTOMATION CONTENT NAVIGATOR GENERAL SETTINGS

The following table describes each general parameter and setting options for automation content navigator.

Table 8.1. Automation content navigator general parameters settings

Parameter	Description	Setting options
ansible-runner-artifact-dir	The directory path to store artifacts generated by ansible-runner.	<p>Default: No default value set</p> <p>CLI: <code>--rad</code> or <code>--ansible-runner-artifact-dir</code></p> <p>ENV: <code>ANSIBLE_NAVIGATOR_ANSIBLE_RUNNER_ARTIFACT_DIR</code></p> <p>Settings file:</p> <pre>ansible-navigator: ansible-runner: artifact-dir:</pre>
ansible-runner-rotate-artifacts-count	Keep ansible-runner artifact directories, for last n runs. If set to 0, artifact directories are not deleted.	<p>Default: No default value set</p> <p>CLI: <code>--rac</code> or <code>--ansible-runner-rotate-artifacts-count</code></p> <p>ENV: <code>ANSIBLE_NAVIGATOR_ANSIBLE_RUNNER_ROTATE_ARTIFACTS_COUNT</code></p> <p>Settings file:</p> <pre>ansible-navigator: ansible-runner: rotate-artifacts-count:</pre>
ansible-runner-timeout	The timeout value after which ansible-runner force stops the execution.	<p>Default: No default value set</p> <p>CLI: <code>--rt</code> or <code>--ansible-runner-timeout</code></p> <p>ENV: <code>ANSIBLE_NAVIGATOR_ANSIBLE_RUNNER_TIMEOUT</code></p> <p>Settings file:</p> <pre>ansible-navigator: ansible-runner: timeout:</pre>

Parameter	Description	Setting options
app	Entry point for automation content navigator.	<p>Choices: collections, config, doc, images, inventory, replay, run or welcome</p> <p>Default: welcome</p> <p>CLI example: ansible-navigator collections</p> <p>ENV: ANSIBLE_NAVIGATOR_APP</p> <p>Settings file:</p> <pre>ansible-navigator: app:</pre>
cmdline	Extra parameters passed to the corresponding command.	<p>Default: No default value</p> <p>CLI: positional</p> <p>ENV: ANSIBLE_NAVIGATOR_CMDLINE</p> <p>Settings file:</p> <pre>ansible-navigator: ansible: cmdline:</pre>
collection-doc-cache-path	The path to the collection doc cache.	<p>Default: \$HOME/.cache/ansible-navigator/collection_doc_cache.db</p> <p>CLI: --cdcp or --collection-doc-cache-path</p> <p>ENV: ANSIBLE_NAVIGATOR_COLLECTION_DOC_CACHE_PATH</p> <p>Settings file:</p> <pre>ansible-navigator: collection-doc-cache-path:</pre>

Parameter	Description	Setting options
container-engine	Specify the container engine (auto=podman then docker).	<p>Choices: auto, podman or docker</p> <p>Default: auto</p> <p>CLI: --ce or --container-engine</p> <p>ENV: ANSIBLE_NAVIGATOR_CONTAINER_ENGINE</p> <p>Settings file:</p> <pre>ansible-navigator: execution-environment: container-engine:</pre>
display-color	Enable the use of color in the display.	<p>Choices: True or False</p> <p>Default: True</p> <p>CLI: --dc or --display-color</p> <p>ENV: NO_COLOR</p> <p>Settings file:</p> <pre>ansible-navigator: color: enable:</pre>
editor-command	Specify the editor used by automation content navigator	<p>Default: <code>* vi +{line_number} {filename}</code></p> <p>CLI: --ecmd or --editor-command</p> <p>ENV: ANSIBLE_NAVIGATOR_EDITOR_COMMAND</p> <p>Settings file:</p> <pre>ansible-navigator: editor: command:</pre>

Parameter	Description	Setting options
editor-console	Specify if the editor is console based.	<p>Choices: True or False</p> <p>Default: True</p> <p>CLI: --econ or --editor-console</p> <p>ENV: ANSIBLE_NAVIGATOR_EDIT OR_CONSOLE</p> <p>Settings file:</p> <pre> ansible-navigator: editor: console: </pre>
execution-environment	Enable or disable the use of an automation execution environment.	<p>Choices: True or False</p> <p>Default: True</p> <p>CLI: --ee or --execution-environment</p> <p>ENV:* ANSIBLE_NAVIGATOR_EXECUTION_ENVIRONMENT</p> <p>Settings file:</p> <pre> ansible-navigator: execution-environment: enabled: </pre>

Parameter	Description	Setting options
execution-environment-image	Specify the name of the automation execution environment image.	<p>Default: quay.io/ansible/ansible-runner:devel</p> <p>CLI: --eei or --execution-environment-image</p> <p>ENV: ANSIBLE_NAVIGATOR_EXECUTION_ENVIRONMENT_IMAGE</p> <p>Settings file:</p> <pre>ansible-navigator: execution-environment: image:</pre>
execution-environment-volume-mounts	Specify volume to be bind mounted within an automation execution environment (--eev /home/user/test:/home/user/test:Z)	<p>Default: No default value set</p> <p>CLI: --eev or --execution-environment-volume-mounts</p> <p>ENV: ANSIBLE_NAVIGATOR_EXECUTION_ENVIRONMENT_VOLUME_MOUNTS</p> <p>Settings file:</p> <pre>ansible-navigator: execution-environment: volume-mounts:</pre>
log-append	Specify if log messages should be appended to an existing log file, otherwise a new log file is created per session.	<p>Choices: True or False</p> <p>Default: True</p> <p>CLI: --la or --log-append</p> <p>ENV: ANSIBLE_NAVIGATOR_LOG_APPEND</p> <p>Settings file:</p> <pre>ansible-navigator: logging: append:</pre>

Parameter	Description	Setting options
log-file	Specify the full path for the automation content navigator log file.	<p>Default: \$PWD/ansible-navigator.log</p> <p>CLI: --lf or --log-file</p> <p>ENV: ANSIBLE_NAVIGATOR_LOG_FILE</p> <p>Settings file:</p> <pre>ansible-navigator: logging: file:</pre>
log-level	Specify the automation content navigator log level.	<p>Choices: debug, info, warning, error or critical</p> <p>Default: warning</p> <p>CLI: --ll or --log-level</p> <p>ENV: ANSIBLE_NAVIGATOR_LOG_LEVEL</p> <p>Settings file:</p> <pre>ansible-navigator: logging: level:</pre>
mode	Specify the user-interface mode.	<p>Choices: stdout or interactive</p> <p>Default: interactive</p> <p>CLI: -m or --mode</p> <p>ENV: ANSIBLE_NAVIGATOR_MODE</p> <p>Settings file:</p> <pre>ansible-navigator: mode:</pre>

Parameter	Description	Setting options
osc4	Enable or disable terminal color changing support with OSC 4.	<p>Choices: True or False</p> <p>Default: True</p> <p>CLI: --osc4 or --no-osc4</p> <p>ENV: ANSIBLE_NAVIGATOR_OSC4</p> <p>Settings file:</p> <pre>ansible-navigator: color: osc4:</pre>
pass-environment-variable	Specify an exiting environment variable to be passed through to and set within the automation execution environment (--penv MY_VAR)	<p>Default: No default value set</p> <p>CLI: --penv or --pass-environment-variable</p> <p>ENV: ANSIBLE_NAVIGATOR_PASS_ENVIRONMENT_VARIABLES</p> <p>Settings file:</p> <pre>ansible-navigator: execution-environment: environment-variables: pass:</pre>
pull-policy	Specify the image pull policy. always - Always pull the image missing - Pull if not locally available never - Never pull the image tag - If the image tag is latest always pull the image, otherwise pull if not locally available	<p>Choices: always, missing, never, or tag</p> <p>Default: tag</p> <p>CLI: --pp or --pull-policy</p> <p>ENV: ANSIBLE_NAVIGATOR_PULL_POLICY</p> <p>Settings file:</p> <pre>ansible-navigator: execution-environment: pull-policy:</pre>

Parameter	Description	Setting options
set-environment-variable	Specify an environment variable and a value to be set within the automation execution environment (--senv MY_VAR=42)	<p>Default: No default value set</p> <p>CLI: --senv or --set-environment-variable</p> <p>ENV: ANSIBLE_NAVIGATOR_SET_ENVIRONMENT_VARIABLES</p> <p>Settings file:</p> <pre>ansible-navigator: execution-environment: environment-variables: set:</pre>

8.3. AUTOMATION CONTENT NAVIGATOR `config` SUBCOMMAND SETTINGS

The following table describes each parameter and setting options for the automation content navigator `config` subcommand.

Table 8.2. Automation content navigator `config` subcommand parameters settings

Parameter	Description	Setting options
config	Specify the path to the Ansible configuration file.	<p>Default: No default value set</p> <p>CLI: -c or --config</p> <p>ENV: ANSIBLE_CONFIG</p> <p>Settings file:</p> <pre>ansible-navigator: ansible: config: path:</pre>

Parameter	Description	Setting options
help-config	Help options for the ansible-config command in stdout mode.	<p>Choices:* True or False</p> <p>Default: False</p> <p>CLI: --hc or --help-config</p> <p>ENV: ANSIBLE_NAVIGATOR_HELP_CONFIG</p> <p>Settings file:</p> <pre>ansible-navigator: help-config:</pre>

8.4. AUTOMATION CONTENT NAVIGATOR **doc** SUBCOMMAND SETTINGS

The following table describes each parameter and setting options for the automation content navigator **doc** subcommand.

Table 8.3. automation content navigator**doc** subcommand parameters settings

Parameter	Description	Setting options
help-doc	Help options for the ansible-doc command in stdout mode.	<p>Choices: True or False</p> <p>Default: False</p> <p>CLI: --hd or --help-doc</p> <p>ENV: ANSIBLE_NAVIGATOR_HELP_DOC</p> <p>Settings file:</p> <pre>ansible-navigator: help-doc:</pre>

Parameter	Description	Setting options
plugin-name	Specify the plugin name.	<p>Default: No default value set</p> <p>CLI: positional</p> <p>ENV: ANSIBLE_NAVIGATOR_PLUGIN_NAME</p> <p>Settings file:</p> <pre>ansible-navigator: documentation: plugin: name:</pre>
plugin-type	Specify the plugin type.	<p>Choices: become, cache, callback, cliconf, connection, httpapi, inventory, lookup, module, netconf, shell, strategy, or vars</p> <p>Default: module</p> <p>CLI: -t or ----type</p> <p>ENV: ANSIBLE_NAVIGATOR_PLUGIN_TYPE</p> <p>Settings file:</p> <pre>ansible-navigator: documentation: plugin: type:</pre>

8.5. AUTOMATION CONTENT NAVIGATOR INVENTORY SUBCOMMAND SETTINGS

The following table describes each parameter and setting options for the automation content navigator **inventory** subcommand.

Table 8.4. Automation content navigator **inventory** subcommand parameters settings

Parameter	Description	Setting options
-----------	-------------	-----------------

Parameter	Description	Setting options
help-inventory	Help options for the ansible-inventory command in stdout mode.	<p>Choices: True or False</p> <p>Default: False</p> <p>CLI: --hi or --help-inventory</p> <p>ENV: ANSIBLE_NAVIGATOR_INVENTORY_DOC</p> <p>Settings file:</p> <pre>ansible-navigator: help-inventory:</pre>
inventory	Specify an inventory file path or comma separated host list.	<p>Default: no default value set</p> <p>CLI: --i or --inventory</p> <p>ENV: ANSIBLE_NAVIGATOR_INVENTORIES</p> <p>Settings file:</p> <pre>ansible-navigator: inventories:</pre>
inventory-column	Specify a host attribute to show in the inventory view.	<p>Default: No default value set</p> <p>CLI: --ic or --inventory-column</p> <p>ENV:*</p> <p>ANSIBLE_NAVIGATOR_INVENTORY_COLUMNS</p> <p>Settings file:</p> <pre>ansible-navigator: inventory-columns:</pre>

8.6. AUTOMATION CONTENT NAVIGATOR REPLAY SUBCOMMAND SETTINGS

The following table describes each parameter and setting options for the automation content navigator **replay** subcommand.

Table 8.5. Automation content navigator **replay** subcommand parameters settings

Parameter	Description	Setting options
playbook-artifact-replay	Specify the path for the playbook artifact to replay.	<p>Default: No default value set</p> <p>CLI: positional</p> <p>ENV: ANSIBLE_NAVIGATOR_PLAYBOOK_ARTIFACT_REPLAY</p> <p>Settings file:</p> <pre>ansible-navigator: playbook-artifact: replay:</pre>

8.7. AUTOMATION CONTENT NAVIGATOR `run` SUBCOMMAND SETTINGS

The following table describes each parameter and setting options for the automation content navigator `run` subcommand.

Table 8.6. Automation content navigator `run` subcommand parameters settings

Parameter	Description	Setting options
playbook-artifact-replay	Specify the path for the playbook artifact to replay.	<p>Default: No default value set</p> <p>CLI: positional</p> <p>ENV: ANSIBLE_NAVIGATOR_PLAYBOOK_ARTIFACT_REPLAY</p> <p>Settings file:</p> <pre>ansible-navigator: playbook-artifact: replay:</pre>

Parameter	Description	Setting options
help-playbook	Help options for the ansible-playbook command in stdout mode.	<p>Choices: True or False</p> <p>Default: False</p> <p>CLI: --hp or --help-playbook</p> <p>ENV: ANSIBLE_NAVIGATOR_HELP_PLAYBOOK</p> <p>Settings file:</p> <pre>ansible-navigator: help-playbook:</pre>
inventory	Specify an inventory file path or comma separated host list.	<p>Default: no default value set</p> <p>CLI: --i or --inventory</p> <p>ENV: ANSIBLE_NAVIGATOR_INVENTORIES</p> <p>Settings file:</p> <pre>ansible-navigator: inventories:</pre>
inventory-column	Specify a host attribute to show in the inventory view.	<p>Default: No default value set</p> <p>CLI: --ic or --inventory-column</p> <p>ENV:*</p> <p>ANSIBLE_NAVIGATOR_INVENTORY_COLUMNSSettings file:</p> <pre>ansible-navigator: inventory-columns:</pre>

Parameter	Description	Setting options
playbook	Specify the playbook name.	<p>Default: No default value set</p> <p>CLI: positional</p> <p>ENV: ANSIBLE_NAVIGATOR_PLAYBOOK</p> <p>Settings file:*</p> <pre>ansible-navigator: ansible: playbook:</pre>
playbook-artifact-enable	<p>Enable or disable the creation of artifacts for completed playbooks.</p> <p>Note: not compatible with --mode stdout when playbooks require user input.</p>	<p>Choices: True or False</p> <p>Default: True</p> <p>CLI: --pae or --playbook-artifact-enable</p> <p>ENV: ANSIBLE_NAVIGATOR_PLAYBOOK_ARTIFACT_ENABLE</p> <p>Settings file:</p> <pre>ansible-navigator: playbook-artifact: enable:</pre>
playbook-artifact-save-as	Specify the name for artifacts created from completed playbooks.	<p>Default: {playbook_dir}/{playbook_name}-artifact-{ts_utc}.json</p> <p>CLI: --pas or --playbook-artifact-save-as</p> <p>ENV: ANSIBLE_NAVIGATOR_PLAYBOOK_ARTIFACT_SAVE_AS</p> <p>Settings file:</p> <pre>ansible-navigator: playbook-artifact: save-as:</pre>

CHAPTER 9. TROUBLESHOOTING ANSIBLE CONTENT WITH AUTOMATION CONTENT NAVIGATOR

As a content creator, you can troubleshoot your Ansible content (collections, automation execution environments, and playbooks) with automation content navigator and interactively troubleshoot the playbook. You can also compare results inside or outside an automation execution environment and troubleshoot any problems.

9.1. REVIEWING PLAYBOOK RESULTS WITH AN AUTOMATION CONTENT NAVIGATOR ARTIFACT FILE

Automation content navigator saves the results of the playbook run in a JSON artifact file. You can use this file to share the playbook results with someone else, save it for security or compliance reasons, or review and troubleshoot later. You only need the artifact file to review the playbook run. You do not need access to the playbook itself or inventory access.

Prerequisites

- A automation content navigator artifact JSON file from a playbook run.

Procedure

- Start automation content navigator with the artifact file.

```
$ ansible-navigator replay simple_playbook_artifact.json
```

1. Review the playbook results that match when the playbook ran.

PLAY NAME	OK	CHANGED	UNREACHABLE	FAILED	SKIPPED	IGNORED	IN PROGRESS	TASK COUNT	PROGRESS
all	12	0	0	0	25	0	0	37	COMPLETE

You can now type the number next to the plays and tasks to step into each to review the results, as you would after executing the playbook.

Additional resources

- [ansible-playbook](#)
- [Ansible playbooks](#)

9.2. FREQUENTLY ASKED QUESTIONS ABOUT AUTOMATION CONTENT NAVIGATOR

Use the following automation content navigator FAQ to help you troubleshoot problems in your environment.

Where should the **ansible.cfg** file go when using an automation execution environment?

The easiest place to have the **ansible.cfg** file is in the project directory next to the playbook. The playbook directory is automatically mounted in the automation execution environment and automation content navigator finds the **ansible.cfg** file there. If the **ansible.cfg** file is in another directory, set the **ANSIBLE_CONFIG** variable, and specify the directory as a custom volume mount. (See automation content navigator settings for **execution-environment-volume-mounts**)

Where should the **ansible.cfg** file go when not using an automation execution environment?

Ansible looks for the **ansible.cfg** in the typical locations when not using an automation execution environment. See [Ansible configuration settings](#) for details.

Where should Ansible collections be placed when using an automation execution environment?

The easiest place to have Ansible collections is in the project directory, in a playbook adjacent collections directory (for example, **ansible-galaxy collections install ansible.utils -p ./collections**). The playbook directory is automatically mounted in the automation execution environment and automation content navigator finds the collections there. Another option is to build the collections into an automation execution environment using Ansible Builder. This helps content creators author playbooks that are production ready, since automation controller supports playbook adjacent collection directories. If the collections are in another directory, set the **ANSIBLE_COLLECTIONS_PATHS** variable and configure a custom volume mount for the directory. (See [Automation content navigator general settings](#) for **execution-environment-volume-mounts**).

Where should Ansible collections be placed when not using an automation execution environment?

When not using an automation execution environment, Ansible looks in the default locations for collections. See the [Using Ansible collections](#) guide.

Why does the playbook hang when `vars_prompt` or `pause/prompt` is used?

By default, automation content navigator runs the playbook in the same manner that automation controller runs the playbook. This helps content creators author playbooks that are production ready. If you cannot avoid the use of **vars_prompt** or **pause/prompt**, disabling **playbook-artifact** creation causes automation content navigator to run the playbook in a manner that is compatible with **ansible-playbook** and allows for user interaction.

Why does automation content navigator change the terminal colors or look terrible?

Automation content navigator queries the terminal for its OSC4 compatibility. OSC4, 10, 11, 104, 110, 111 indicate the terminal supports color changing and reverting. It is possible that the terminal is misrepresenting its ability. You can disable OSC4 detection by setting **--osc4 false**. (See [Automation content navigator general settings](#) for how to handle this with an environment variable or in the settings file).

How can I change the colors used by automation content navigator?

Use **--osc4 false** to force automation content navigator to use the terminal defined colors. (See [Automation content navigator general settings](#) for how to handle this with an environment variable or in the settings file).

What is with all these `site-artifact-2021-06-02T16:02:33.911259+00:00.json` files in the playbook directory?

Automation content navigator creates a playbook artifact for every playbook run. These can be helpful for reviewing the outcome of automation after it is complete, sharing and troubleshooting with a colleague, or keeping for compliance or change-control purposes. The playbook artifact file has the detailed information about every play and task, and the **stdout** from the playbook run. You can review playbook artifacts with **ansible-navigator replay <filename>** or **:replay <filename>** while in an automation content navigator session. You can review all playbook artifacts with both **--mode stdout** and **--mode interactive**, depending on the required view. You can disable playbook artifacts writing and the default file naming convention. (See [Automation content navigator general settings](#) for how to handle this with an environment variable or in the settings file).

Why does `vi` open when I use `:open`?

Automation content navigator opens anything showing in the terminal in the default editor. The default is set to either **vi +{line_number} {filename}** or the current value of the **EDITOR** environment variable. Related to this is the **editor-console** setting which indicates if the editor is console or terminal based. Here are examples of alternate settings that might be useful:

```
# emacs
ansible-navigator:
  editor:
```

```
command: emacs -nw +{line_number} {filename}  
console: true
```

```
# vscode  
ansible-navigator:  
editor:  
  command: code -g {filename}:{line_number}  
  console: false
```

```
#pycharm  
ansible-navigator:  
editor:  
  command: charm --line {line_number} {filename}  
  console: false
```

What is the order in which configuration settings are applied?

The automation content navigator configuration system pulls in settings from various sources and applies them hierarchically in the following order (where the last applied changes are the most prevalent):

1. Default internal values
2. Values from a settings file
3. Values from environment variables
4. Flags and arguments specified on the command line
5. While issuing `:` commands within the text-based user interface

Something did not work, how can I troubleshoot it?

Automation content navigator has reasonable logging messages. You can enable **debug** logging with **--log-level debug**. If you think you might have found a bug, log an issue and include the details from the log file.