Red Hat Ansible Automation Platform 2.2

Red Hat Ansible Automation Platform Creator ガイド

このガイドは、自動化用のコンテンツを作成する際に Ansible を使用する方法を学びたい開発者を対象としています。

Last Updated: 2023-05-16

Red Hat Ansible Automation Platform 2.2 Red Hat Ansible Automation Platform Creator ガイド

このガイドは、自動化用のコンテンツを作成する際に Ansible を使用する方法を学びたい開発者を対象としています。

法律上の通知

概要

フィードバックの提供: このドキュメントを改善するための提案がある場合、またはエラーを見つけた場合は、テクニカルサポート () に連絡し、Docs コンポーネントを使用して Ansible Automation Platform Jira プロジェクトで issue を作成してください。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

第1章 はじめに

自動化実行環境を使用した Red Hat Ansible Automation Platform 内のコンテンツの自動化

実行環境は、再現可能で、移植可能で、一貫性があり、共有可能なコンテナーイメージとして使用できます。Ansible Automation Platform ジョブのランタイム環境のすべての依存関係を、システムの依存関係、Python の依存関係、Ansible のバージョン、コレクションの形式で Ansible のコンテンツから制御します。

第2章 コンテンツ作成者のワークフローおよび自動化実行環境の概要

2.1. コンテンツワークフローについて

Red Hat Ansible Automation Platform 2.0 より前のバージョンでは、自動化コンテンツ開発者は非常に多くの Python 仮想環境を必要としていたため、それを管理するために独自の自動化が必要でした。このレベルの複雑性を軽減するため、Ansible Automation Platform 2.0 は仮想環境から離れ、代わりに自動化実行環境と呼ばれるコンテナーを使用しています。コンテナーは構築と管理が簡単で、チーム間や組織間での共有が容易になります。

自動化コントローラーが自動化実行環境の使用に移行するにつれて、自動化コンテンツナビゲーターや Ansible Builder などのツールを使用すると、このような自動化実行環境を独自の開発システム内でローカルに利用できるようになります。

関連情報

- Automation コンテンツナビゲーターの使用の詳細は、Ansible Navigator クリエイターガイド を参照してください。
- Ansible Builder の詳細は、実行環境の作成と消費 を参照してください。

2.2. アーキテクチャーの概要

次の一覧は、Ansible Automation Platform 2.0 で利用可能なツールの配置と使用、およびその使用方法を示しています。

- 自動化コンテンツナビゲーターだけが、現在 Ansible Automation Platform 1.2 で使用できます。
- 自動化コンテンツナビゲーターと、ダウンロードした自動化実行環境は、ラップトップまたはワークステーションで直接使用されます。
- 自動化コンテンツナビゲーター、ダウンロードされた自動化実行環境、および自動化コントローラーをローカルからリモートにプッシュまたは実行します。
- 自動化コンテンツナビゲーター、自動化コントローラー、Ansible Builder、およびレイヤー化されたカスタム EE は、自動化ジョブの実行方法に使用されるコンテンツをさらに細かく制御できます

第3章 ANSIBLE の概念について

自動化開発者は、以下の Ansible の概念を確認し、Ansible 開発プロジェクトの開始前に、成功した Ansible Playbook と自動化実行環境を作成します。

3.1. 前提条件

- Ansible がインストールされている。Ansible のインストールに関する情報は、Ansible ドキュメントの Ansible のインストール を参照してください。

3.2. ANSIBLE PLAYBOOK について

Playbook は、特定の人間が判読可能な命令のセットを含む YAML で書かれたファイルです。これには、単一ターゲットまたはターゲットグループで実行するように送信されます。

Playbook を使用して、リモートマシンの設定とデプロイメントを管理したり、ローリング更新を含む多層ロールアウトを順序付けたりすることができます。Playbook を使用して、アクションを他のホストに委譲し、途中で監視サーバーやロードバランサーと対話します。作成した Playbook は、自動化のために企業全体で繰り返し使用できます。

3.3. ANSIBLE ロールについて

ロールは、自動化コンテンツをバンドルするだけでなく、既知のファイル構造を利用して関連する変数、ファイル、タスク、ハンドラー、およびその他のアーティファクトを自動的に読み込む Ansible の方法です。何百ものタスクで巨大な Playbook を作成する代わりに、ロールを使用して、タスクをより小さく、より個別のコンポーザブルな作業単位に分割することができます。

インフラストラクチャーのプロビジョニング、アプリケーションのデプロイ、および Ansible Galaxy で各タスクを実行するすべてのタスクにロールがあります。**Type** で検索にフィルターを設定し、**Role** を選択します。対象のロールを見つけたら、Ansible に同梱されている **ansible-galaxy** コマンドを使用してダウンロードできます。

```
$ ansible-galaxy role install username.rolename
```

3.4. コンテンツコレクションについて

Ansible Content Collection は、自動化用の使いやすいツールキットです。Playbook、ロール、モジュール、プラグインなど、複数の種類のコンテンツがすべて 1 か所に含まれています。以下の図は、コレクションの基本構造を示しています。

```
collection/
├── docs/
├── galaxy.yml
├── meta/
│   └── runtime.yml
├── plugins/
│   ├── modules/
│   │   └── module1.py
│   ├── inventory/
│   ├── lookup/
│   ├── filter/
│   └── .../
├── README.md
```

```
├── roles/
│   ├── role1/
│   ├── role2/
│   └── .../
├── playbooks/
│   ├── files/
│   ├── vars/
│   ├── templates/
│   ├── playbook1.yml
│   └── tasks/
└── tests/
    ├── integration/
    └── unit/
```

Red Hat Ansible Automation Platform では、自動化ハブは、Ansible 認定コンテンツコレクションのソースとして機能します。

3.5. 実行環境について

自動化実行環境は、Ansible コントロールノードとして機能する一貫性のある共有可能なコンテナーイメージです。自動化実行環境は、外部の依存関係のある Ansible コンテンツを共有する際の課題を軽減します。

自動化実行環境には、以下が含まれます。

- Ansible Core
- Ansible Runner
- Ansible コレクション
- Python ライブラリー
- システムの依存関係
- カスタムのユーザーニーズ

Ansible Builder を使用して自動化実行環境を定義および作成することができます。

関連情報

- Ansible Builder の詳細は、実行環境の作成と消費 を参照してください。

第4章 ツールおよびコンポーネント

自動化実行環境の作成に使用する Red Hat Ansible Automation Platform ツールとコンポーネントの詳細をご覧ください。

4.1. ANSIBLE BUILDER について

Ansible Builder は、さまざまな Ansible Collection で定義されたメタデータ、またはユーザーが定義したメタデータを使用して、自動化実行環境を構築するプロセスを自動化するコマンドラインツールです。

Ansible Builder が開発される前に、Red Hat Ansible Automation Platform ユーザーは、必要なすべての依存関係がインストールされているカスタムの仮想環境またはコンテナーの作成時に、依存関係の問題およびエラーを実行することができました。

Ansible Builder を使用すると、コレクション、サードパーティーの Python 要件、システムレベルパッケージなど、自動化実行環境に含めるコンテンツを指定する、カスタマイズ可能な自動化実行環境定義ファイルを簡単に作成できます。これにより、ジョブの実行に必要な要件と依存関係をすべて満たすことができます。

注記

Red Hat は現在、自動化の実行環境を構築する際に独自のコンテナーイメージを提供することを選択したユーザーをサポートしていません。

4.2. 自動化コンテンツナビゲーターへの使用

自動化コンテンツナビゲーターは、テキストベースのユーザーインターフェイスを備えたコマンドラインのコンテンツクリエーターに焦点を当てたツールです。自動化コンテンツナビゲーターを使用して、以下を実行できます。

- ジョブおよび Playbook を起動し、監視します。
- 保存され、完成した Playbook とジョブ実行アーティファクトを JSON 形式で共有します。
- 自動化実行環境を閲覧およびイントロスペクションします。
- ファイルベースのインベントリーを参照します。
- Ansible モジュールのドキュメントをレンダリングし、Playbook で使用できるサンプルを展開します。

自動化実行環境などのコンテナーでは、**ansible-playbook** などの Ansible コアの一部である Ansible コマンドを実行できません。自動化実行環境と互換性がある Ansible CLI コマンドセットが含まれる、自動化コンテンツナビゲーターを使用します。Automation コンテンツナビゲーターでは、テキストベースのユーザーインターフェイス内にさらに詳細な出力も含まれます。

4.3. AUTOMATION HUB について

Automation Hub は、Red Hat サブスクリプションをお持ちのお客様が、Red Hat および弊社のテクノロジーパートナーがサポートするコンテンツをすばやく見つけて使用できる場所を提供し、最も要求の厳しい環境にさらなる安心を提供します。

大まかには、Automation Hub はプログラムに参加し認定されサポートされるコンテンツを提供する全パートナーの概要を説明します。

中央のビューから、ユーザーは各パートナーをより深く掘り下げ、コレクションをチェックアウトできます。

さらに、利用可能なすべてのコレクションの検索可能な概要が利用可能です。

4.4. ANSIBLE コマンドラインインターフェイスについて

コマンドラインで Ansible を使用すると、非常に頻繁に繰り返されないタスクを実行するために便利な方法になります。タスクを繰り返す場合は、Playbook を作成することが推奨されます。

コマンドラインにおける Ansible 用のアドホックコマンドは、以下の構造に従います。

```
$ ansible [pattern] -m [module] -a "[module options]"
```

4.5. 関連情報

- Ansible をコマンドラインツールとして使用する方法は、Ansible の **ユーザーガイド** で、コマンドラインツールの使用 を参照してください。
- 自動化ハブにコンテンツをアップロードするには、Ansible Automation Platform 製品ドキュメントの 自動化ハブへのコンテンツのアップロード を参照してください。

第5章 開発環境の設定

このセクションの手順に従って、自動化実行環境を作成するための開発環境をセットアップできます。

5.1. ANSIBLE BUILDER のインストール

Red Hat Subscription Management (RHSM) を使用して Ansible Builder をインストールし、Red Hat Ansible Automation Platform サブスクリプションをアタッチできます。Red Hat Ansible Automation Platform サブスクリプションをアタッチ すると、**ansible-builder** のインストールに必要なサブスクリプションのみのリソースにアクセスできます。サブスクリプションをアタッチすると、**ansible-builder** に必要なリポジトリーが自動的に有効になります。

注記

ansible-builder をインストールする前に、有効なサブスクリプションがホストにアタッチされている必要があります。

手順

1. ターミナルで次のコマンドを実行して、Ansible Automation Platform リポジトリーをアクティブ化します。

```
# dnf config-manager --enable ansible-automation-platform-2.2-for-rhel-8-x86_64-rpms
```

2. 次に、次のコマンドを入力して Ansible Builder をインストールします。

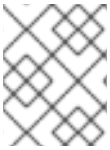
```
# dnf install ansible-builder
```

5.2. RPM からの RHEL への自動化コンテンツナビゲーターのインストール

RPM から、Red Hat Enterprise Linux (RHEL) に自動化コンテンツナビゲーターをインストールできます。

前提条件

- RHEL 8 以降をインストールしている。
- Red Hat Subscription Manager でシステムを登録している。



注記

現在の Red Hat Ansible Automation Platform 環境に一致するナビゲーターのみをインストールしてください。

手順

1. Red Hat Ansible Automation Platform SKU を割り当てます。

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

2. 次のコマンドを使用して、Automation コンテンツナビゲーターをインストールします。

```
# dnf install --enablerepo=ansible-automation-platform-2.2-for-rhel-8-x86_64-rpms ansible-navigator
```

検証

- 自動化コンテンツナビゲーターのインストールを確認します。

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

以下の例は、インストールの成功を示しています。

```
$ 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
```

5.3. ベース自動化実行環境のダウンロード

AAP 2.0 に同梱されるベースイメージは、Red Hat Ecosystem Catalog (registry.redhat.io) でホストされているものです。

前提条件

- 有効な Red Hat Ansible Automation Platform サブスクリプションがある。

手順

1. registry.redhat.io にログインします。

```
$ podman login registry.redhat.com
```

2. レジストリーからベースイメージをプルします。

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

第6章 コンテンツの作成

Red Hat Ansible Automation Platform で使用するコンテンツの開発に関する詳細は、**作成者ガイド** でこのセクションに関するガイドラインを参照してしてください。

6.1. PLAYBOOK の作成

Playbook には、1つ以上のプレイが含まれます。基本的なプレイには以下のセクションが含まれます。

- Name: Playbook の全体的な機能について簡単に説明し、全ユーザーに対して読み取り可能かつ整理された状態を維持するのに役立ちます。
- Hosts: Ansible が実行するターゲットを特定します。
- become ステートメント: この任意のステートメントは、become プラグイン (例: **sudo**、**su**、**pfexec**、**doas**、**pbrun**、**dzdo**、**ksu**) を使用して権限のエスカレーションを有効にするために **true**/**yes** に設定できます。
- tasks: これは、プレイの各ホストに対して実行されるアクションのリストです。

Playbook の例

```
- 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
```

6.2. コレクションの作成

Ansible Galaxy CLI ツールを使用して、独自のコレクションをローカルで作成できます。コレクション固有のコマンドはすべて、**collection** サブコマンドを使用してアクティベートできます。

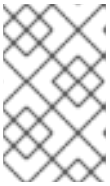
前提条件

- 開発環境に Ansible バージョン 2.9 以降がインストールされている。

手順

1. 端末で、名前空間の root ディレクトリーの場所に移動します。分かりやすくするため、これは COLLECTIONS_PATH のパスにすることが推奨されますが、必須ではありません。
2. 以下のコマンドを実行して、**my_namespace** と **my_collection_name** を選択した値に置き換えます。

```
$ ansible-galaxy collection init <my_namespace>.<my_collection_name>
```



注記

galaxy.ansible.com または cloud.redhat.com/ansible/automation-hub の My Content タブの下で、名前空間にアップロードする適切なパーミッションがあることを確認してください。

上記のコマンドは、上記の名前空間引数から名前が付けられたディレクトリーが存在しない場合は作成し、そのディレクトリーに Collection 名を指定してディレクトリーを作成します。そのディレクトリー内では、デフォルトまたはスケルトン Collection になります。ここでは、ロールまたはプラグインを追加し、独自の Collection 開発に取り掛かることができます。

実行環境に関連して、コレクション開発者は Ansible Builder で適切なメタデータを提供することで、コンテンツの要件を宣言できます。

Collection からの要件は、以下の方法で認識できます。

- **meta/execution-environment.yml** ファイルが Python または **bindep** 要件ファイル、もしくはその両方を参照します。
- Python の依存関係に関する情報を含む **requirements.txt** という名前のファイルは、Collection のルートディレクトリーにある場合があります。
- システムレベルの依存関係を含む **bindep.txt** という名前のファイルは、Collection のルートレベルに存在する場合があります。
- これらのファイルのいずれかが Collection の **build_ignore** にある場合、このセクションはビルドアーティファクトに含めるべきではないファイルまたはディレクトリーにフィルターを設定するために使用されるため、Ansible Builder はこれらを取得しません。

コレクションメンテナーは、introspect コマンドを使用して、ansible-builder が想定する要件を認識していることを確認できます。

```
$ ansible-builder introspect --sanitize ~/.ansible/collections/
```

関連情報

- コレクションの作成に関する詳細は、Ansible **開発者ガイド**の コレクションの作成 を参照してください。

6.3. ロールの作成

Ansible Galaxy CLI ツールを使用してロールを作成できます。ロール固有のコマンドは、**roles** サブコマンドからアクセスできます。

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

コレクション外部のスタンドアロンロールは引き続きサポートされますが、Ansible Automation Platform が提供するすべての機能を活用するには、新しいロールをコレクション内にを作成する必要があります。

手順

1. ターミナルで、コレクション内の **roles** ディレクトリーに移動します。
2. 以前に作成したコレクション内に **role_name** という名前のロールを作成します。

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

コレクションには、**roles** ディレクトリー内に **my_role** という名前のロールが含まれるようになりました。

```
~/.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. カスタムロールのスケルトンディレクトリーは、**--role-skeleton** 引数を使用して指定できます。これにより、組織はニーズに合わせて新しいロールの標準化されたテンプレートを作成することができます。

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

これにより ~/**role_skeleton** の内容を **my_role** にコピーして、**my_role** という名前のロールが作成されます。**role_skeleton** の内容は、ロールディレクトリー内で有効なすべてのファイルまたはフォルダーになります。

関連情報

- ロールの作成に関する詳細は、Ansible Galaxy ドキュメントの ロールの作成 を参照してください。

6.4. 自動化実行環境の作成

自動化実行環境定義ファイルでは、以下を指定します。

- Ansible バージョン
- Python バージョン (デフォルトはシステム Python)
- 必要な Python ライブラリーのセット
- ゼロ以上の Content Collection (任意)
- その特定の Collection の Python 依存関係

環境に Collection のセットを指定する概念は、依存関係を解決し、インストールすることです。Collection 自体は、自動化実行環境を生成するマシンにインストールする必要はありません。

自動化実行環境は、この定義から構築され、コンテナーイメージが作成されます。これらのイメージの作成手順については、Ansible Builder のドキュメントを参照してください。

第7章 既存のコンテンツの移行

以下のセクションで、**awx-manage** コマンドを使用して、Red Hat Ansible Automation Platform 2.0 および自動化コントローラー 4.0 にアップグレードした後に、移行プロセスの追加手順を支援する方法を説明します。また、Ansible のバージョン間での移行についても確認してください。

7.1. 仮想環境を自動化実行環境に移行

Red Hat Ansible Automation Platform 2.0 および自動化コントローラー 4.0 にアップグレードしたら、移行プロセスの追加手順について、以下のセクションを使用します。

7.1.1. カスタムの仮想環境の構築

awx-manage コマンドを使用して、自動化コントローラーインスタンスの仮想環境を一覧表示できます。

手順

1. 自動コントローラーインスタンスに SSH 接続して実行します。

```
$ awx-manage list_custom_venvs
```

検出された仮想環境の一覧が表示されます。

```
# Discovered virtual environments:
/var/lib/awx/venv/testing
/var/lib/venv/new_env

To export the contents of a virtual environment, re-run while supplying the path as an argument:
awx-manage export_custom_venv /path/to/venv
```

7.1.2. カスタムの仮想環境に関連付けられたオブジェクトの表示

awx-manage コマンドを使用して、カスタムの仮想環境に関連付けられた組織、ジョブ、およびインベントリーソースを表示します。

手順

1. 自動コントローラーインスタンスに SSH 接続して実行します。

```
$ awx-manage custom_venv_associations /path/to/venv
```

関連付けられたオブジェクトの一覧が表示されます。

```
inventory_sources:
- id: 15
  name: celery
job_templates:
- id: 9
  name: Demo Job Template @ 2:40:47 PM
- id: 13
  name: elephant
```

```
organizations
- id: 3
  name: alternating_bongo_meow
- id: 1
  name: Default
projects: []
```

7.1.3. エクスポートするカスタムの仮想環境の選択

awx-manage export_custom_venv コマンドを使用して、エクスポートするカスタムの仮想環境を選択します。

手順

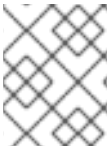
1. 自動コントローラーインスタンスに SSH 接続して実行します。

```
$ awx-manage export_custom_venv /path/to/venv
```

このコマンドにより、指定した仮想環境に含まれる **pip freeze** が表示されます。この情報は、Ansible Builder の **requirements.txt** ファイルにコピーして、新しい自動化実行環境イメージを作成するのに使用できます。

```
numpy==1.20.2
pandas==1.2.4
python-dateutil==2.8.1
pytz==2021.1
six==1.16.0

To list all available custom virtual environments run:
awx-manage list_custom_venvs
```



注記

awx-manage list_custom_venvs を実行して出力を減らしたときに **-q** フラグを渡します。

7.2. ANSIBLE CORE バージョン間の移行

Ansible Core のバージョン間で移行するには、Playbook、プラグイン、その他の Ansible インフラストラクチャーの他の部分を更新して、最新バージョンで機能するようにする必要があります。このプロセスでは、変更が Ansible Core の連続する各バージョンに対して行われる更新に対して検証されている必要があります。Ansible 2.9 から Ansible 2.11 に移行する場合は、最初に Ansible 2.10 の要件を満たしていることを確認し、そこから 2.11 に更新する必要があります。

7.2.1. Ansible 移植ガイド

Ansible **移植ガイド** は、連続する Ansible バージョン間での動作変更に関する情報を提供する一連のドキュメントです。Ansible のバージョンから新しいバージョンに移行する場合は、ガイドを参照してください。

7.2.2. 関連情報

- Ansible 2.8 から Ansible 2.9 における動作の変更については、Ansible 2.9 を参照してください。
- Ansible 2.9 から Ansible 2.10 における動作の変更については、Ansible 2.10 を参照してください。

第8章 AUTOMATION CONTENT NAVIGATOR によるコンテンツの実行

これで、自動化実行環境が構築されました。Ansible コンテンツナビゲーターを使用して、自動化コントローラーが実行するのと同じ方法でコンテンツが実行することを検証できます。

8.1. 自動化コンテンツナビゲーターを使用した ANSIBLE PLAYBOOK の実行

コンテンツ作成者は、自動化コンテンツナビゲーターを使用して Ansible Playbook を実行し、各プレイとタスクの結果をインタラクティブに調べて、Playbook を検証またはトラブルシューティングできます。また、実行環境内で Ansible Playbook を実行すれば、実行環境がなくても、問題を比較してトラブルシューティングできます。

8.1.1. 自動化コンテンツナビゲーターからの Playbook の実行

自動化ナビゲーターのテキストベースのユーザーインターフェイスで Ansible Playbook を実行して、タスクの実行を追跡し、各タスクの結果を詳しく調べることができます。

前提条件

- Playbook
- **localhost** または inventory プラグインを使用しない場合は有効なインベントリーファイル。

手順

1. 自動化コンテンツナビゲーターを起動します。

```
$ ansible-navigator
```

2. Playbook を実行します。

```
$ :run
```

3. オプション: **ansible-navigator run simple-playbook.yml -i inventory.yml** を入力して Playbook を実行します。

4. インベントリーおよびその他のコマンドラインパラメーターを確認または追加します。

```
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 キーを押して **Submit** に移動し、Enter キーを押します。実行中のタスクが表示されるはずです。

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

6. プレイ結果にステップインするプレイの横に番号を入力します。9 を超える数字の場合は **:<number>** と入力します。

```
   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
```

自動化コンテンツナビゲーターで色を有効にしている場合は、失敗したタスクが赤で表示されることに注意してください。

7. タスクの結果を確認するタスクの横に番号を入力します。9 を超える番号の場合は **:<number>** と入力します。

```
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. オプション::**doc** と入力し、トラブルシューティングを支援するタスクで使用されるモジュールまたはプラグインのドキュメントを起動します。

```
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 ...>
```

関連情報

- ansible-playbook.

- Ansible Playbook の概要

8.1.2. 自動化コンテンツナビゲーターアーティファクトファイルを使用した Playbook の結果の確認

Automation コンテンツナビゲーターは、JSON アーティファクトファイルに Playbook 実行の結果を保存します。このファイルを使用して、Playbook の結果を他の人と共有したり、セキュリティーやコンプライアンスの理由で保存したり、後で確認してトラブルシューティングしたりできます。Playbook の実行を確認するには、アーティファクトファイルのみが必要です。Playbook 自体またはインベントリーアクセスにアクセスする必要はありません。

前提条件

- Playbook 実行からの自動化コンテンツナビゲーターアーティファクトの JSON ファイル。

手順

- アーティファクトファイルを使用して自動化コンテンツナビゲーターを起動します。

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

1. Playbook の実行時に一致する Playbook の結果を確認します。

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

これで、Playbook を実行した後と同じように、プレイとタスクの横に番号を入力して、それぞれにステップインして結果を確認できるようになりました。

関連情報

- ansible-playbook.
- Ansible Playbook の概要

第9章 まとめ

これで、特定の自動化のニーズに対応する自動化実行環境をカスタマイズし、コンテナーレジストリーで共有し、使用できるようになりました。