Red Hat Ansible Automation Platform 2.4

Ansible 自動化コンテンツの開発

自動化ジョブを実行するための Ansible 自動化コンテンツを開発する

Last Updated: 2024-09-10

Red Hat Ansible Automation Platform 2.4 Ansible 自動化コンテンツの開発

自動化ジョブを実行するための Ansible 自動化コンテンツを開発する

法律上の通知

概要

このガイドでは、Ansible 自動化コンテンツを開発する方法と、それを使用して Red Hat Ansible Automation Platform から自動化ジョブを実行する方法を説明します。

目次

はじめに **3**

RED HAT ドキュメントへのフィードバック (英語のみ) **4**

第1章 ANSIBLE 開発者ツール **5**
1.1. ANSIBLE 開発者ツールコンポーネント 5

第2章 自動化コンテンツ開発のワークフロー **6**
2.1. ワークフロー 6

第3章 ANSIBLE 開発者ツールのインストール **7**
3.1. 要件 7
3.2. VS コードのインストール 7
3.3. VS CODE ANSIBLE エクステンションのインストール 7
3.4. ANSIBLE エクステンションの設定 7
3.5. VS CODE 内のコンテナーに ANSIBLE 開発ツールをインストールする 8
3.6. RHEL 上のパッケージから ANSIBLE 開発ツールをインストールする 10

第4章 PLAYBOOK プロジェクトの作成 **13**
4.1. PLAYBOOK プロジェクトのスキャフォールディング 13

第5章 ANSIBLE 開発者ツールで PLAYBOOK を作成して実行する **15**
5.1. 最初の PLAYBOOK の作成 15
5.2. PLAYBOOK の検査 15
5.3. PLAYBOOK を実行する 16
5.4. PLAYBOOK のデバッグ 19
5.5. PLAYBOOK のテスト 19

第6章 ANSIBLE AUTOMATION PLATFORM での PLAYBOOK の公開と実行 **21**
6.1. SCM にプロジェクトを保存する 21
6.2. ANSIBLE AUTOMATION PLATFORM で PLAYBOOK を実行する 21

はじめに

Red Hat Ansible Automation Platform に興味をお持ちいただきありがとうございます。Ansible Automation Platform は、Ansible を装備した環境に、制御、ナレッジ、委譲の機能を追加して、チームが複雑かつ複数層のデプロイメントを管理できるように支援する商用サービスです。

このガイドでは、Ansible 自動化コンテンツを開発する方法と、それを使用して Red Hat Ansible Automation Platform から自動化ジョブを実行する方法を説明します。このガイドの更新により、Ansible Automation Platform の最新リリースの情報が追加されました。

RED HAT ドキュメントへのフィードバック (英語のみ)

このドキュメントの改善に関するご意見がある場合や、エラーを発見した場合は、https://access.redhat.com から Technical Support チームに連絡してください。

重要

Ansible plug-ins for Red Hat Developer Hub はテクノロジープレビュー機能としてのみご利用いただけます。テクノロジープレビュー機能は、Red Hat 製品のサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではないことがあります。Red Hat では、実稼働環境での使用を推奨していません。テクノロジープレビューの機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

第1章 ANSIBLE 開発者ツール

Ansible 開発者ツール (**ansible-dev-tools**) は、Ansible Automation Platform で提供されるツールスイートで、自動化の作成者が Playbook プロジェクト、実行環境、コレクションを作成、テスト、デプロイする際に役立ちます。

Red Hat の Ansible VS Code エクステンションには、ほとんどの Ansible 開発者ツールが統合されており、VS Code ユーザーインターフェイスからこれらのツールを使用できます。

Playbook のローカル開発、ローカルテスト、および CI パイプライン (リンティングとテスト) 中に、Ansible 開発者ツールを使用します。

このドキュメントでは、Ansible 開発者ツールを使用して、プロジェクト内で再利用できる Playbook とロールを含む Playbook プロジェクトを作成する方法を説明します。また、Playbook をテストし、{AAP} インスタンスにプロジェクトをデプロイして自動化ジョブで Playbook を使用できるようにする方法についても説明します。

1.1. ANSIBLE 開発者ツールコンポーネント

Ansible エクステンションをインストールすると、一部の Ansible 開発ツールは VS Code UI から、残りはコマンドラインから操作できます。VS Code は、Linux、Mac、Windows で利用できる無料のオープンソースコードエディターです。

Ansible VS Code エクステンション

これは Ansible Automation Platform RPM パッケージには含まれていませんが、自動化作成ワークフローの不可欠な構成要素です。VS Code UI から、Ansible 開発者ツールを使用して次のタスクを実行できます。

- Playbook プロジェクトまたはコレクションのスキャフォールディングディレクトリー。
- 構文の強調表示とオートコンプリートを利用して Playbook を作成します。
- リンターを使用して Playbook をデバッグします。
- **ansible-playbook** を使用して、Ansible Core で Playbook を実行します。
- **ansible-navigator** を使用して、実行環境で Playbook を実行します。

VS Code エクステンションから、Red Hat Ansible Lightspeed with IBM watsonx Code Assistant に接続することもできます。

コマンドライン Ansible 開発ツール

VS Code のターミナルを含め、コマンドラインから Ansible 開発者ツールを使用して次のタスクを実行できます。

- 実行環境を作成します。
- Playbook、ロール、モジュール、プラグイン、コレクションをテストします。

第2章 自動化コンテンツ開発のワークフロー

2.1. ワークフロー

2.1.1. Create

作成段階では、VS Code を使用してローカルで新しい Playbook プロジェクトを作成します。一般的なワークフローは次のとおりです。

1. VS Code に Ansible エクステンションをインストールして実行します。
2. VS Code から Playbook プロジェクトをスキャフォールディングします。
3. プロジェクトに Playbook ファイルを追加し、VS Code で編集します。

2.1.2. テスト

1. **ansible-lint** を利用して Playbook をデバッグします。
2. ローカル環境が Ansible Automation Platform 上の環境を複製するように、自動化実行環境を選択または作成します。
3. **ansible-playbook** を使用するか、実行環境で **ansible-navigator** を使用して、VS Code から Playbook を実行します。
4. 実稼働環境を複製した実行環境で Playbook を実行してテストします。

2.1.3. デプロイ

1. Playbook プロジェクトをソース管理リポジトリーにプッシュします。
2. Ansible Automation Platform で認証情報を設定し、ソース管理リポジトリーからプルして、Playbook リポジトリーのプロジェクトを作成します。
3. 実行環境を作成した場合は、それを Private Automation Hub にプッシュします。
4. Ansible Automation Platform でプロジェクトから Playbook を実行するジョブテンプレートを作成し、使用する実行環境を指定します。

第3章 ANSIBLE 開発者ツールのインストール

Red Hat は、Ansible 開発者ツールをインストールするための 2 つのオプションを提供しています。どちらのオプションでも、Ansible エクステンションが追加された VS Code (Visual Studio Code) が必要です。

- VS Code 内で実行されている RHEL コンテナーへのインストール。このオプションは、MacOS、Windows、Linux システムにインストールできます。
- RPM (Red Hat Package Manager) パッケージを使用した。ローカル RHEL システムへのインストール。

3.1. 要件

Ansible 開発ツールには Python 3.10 以降が必要です。

3.2. VS コードのインストール

VS Code をインストールするには、Visual Studio Code ドキュメントの Download Visual Studio Code page の指示に従ってください。

3.3. VS CODE ANSIBLE エクステンションのインストール

Ansible エクステンションは、VS Code に Ansible の言語サポートを追加します。自動化コンテンツの作成と実行を容易にするために、Ansible 開発者ツールが組み込まれています。

Ansible エクステンションの詳細な説明については、Visual Studio Code Marketplace を参照してください。

エクステンションの操作に関するチュートリアルは、Learning path - Getting Started with the Ansible VS Code Extension を参照してください。

Ansible VS Code エクステンションをインストールするには、以下を実行します。

1. VS Code を開きます。
2. アクティビティーバーの **Extensions** ( ) アイコンをクリックするか、**View → Extensions** をクリックして **Extensions** ビューを表示します。
3. **Extensions** ビューの検索フィールドに **Ansible Red Hat** と入力します。
4. Ansible エクステンションを選択し、**Install** をクリックします。

検証

Ansible エクステンションが有効になっている場合は、VS Code で **.yml** ファイルを作成するか開きます。空のファイルも使用できます。.yml ファイルで識別される言語は Ansible です。ステータスバーに表示されます。

ファイルの言語が Ansible として認識されると、Ansible エクステンションはオートコンプリート、ホバー、診断、goto などの機能を提供します。

3.4. ANSIBLE エクステンションの設定

Ansible エクステンションは複数の設定オプションをサポートしています。

エクステンションの設定は、ユーザーレベル、ワークスペースレベル、または特定のディレクトリーで設定できます。ユーザーベースの設定は、開かれている VS Code のすべてのインスタンスに対し、グローバルに適用されます。ワークスペース設定はワークスペース内に保存され、現在のワークスペースを開いたときにのみ適用されます。

ワークスペースを設定すると、以下の点で便利です。

- Playbook プロジェクトに固有の設定を定義して維持すると、他の作業の優先設定を変更することなく、個々のプロジェクトに合わせて Ansible 開発環境をカスタマイズできます。Python プロジェクト、Ansible プロジェクト、C++ プロジェクトごとに異なる設定が可能で、それぞれのスタックに合わせて最適化されているため、プロジェクトを切り替えるたびに手動で再設定する必要はありません。
- チームと共有するプロジェクトのバージョン管理をセットアップするときにワークスペース設定を含めると、全員がそのプロジェクトに対して同じ設定を使用します。

手順

1. Ansible エクステンションの設定を開きます。
   a. アクティビティーバーで、'Extensions' アイコンをクリックします。
   b. Ansible エクステンションを選択し、'歯車' アイコン、**Extension Settings** の順にクリックしてエクステンション設定を表示します。
   または、**Code → Settings → Settings** をクリックして **Settings** ページを開きます。
   c. 検索バーに **Ansible** と入力してエクステンションの設定を表示します。
2. 現在の VS Code ワークスペースの設定を行うには、**Workspace** タブを選択します。
3. Ansible エクステンションの設定は事前に入力されています。要件に合わせてこ設定を変更します。
   - ansible-lint を有効にするには、**Ansible → Validation → Lint: Enabled** ボックスをオンにします。
   - 実行環境を使用するには、**Ansible Execution Environment: Enabled** ボックスをオンにします。
   - **Ansible > Execution Environment: image**フィールドで使用する実行環境イメージを指定します。
   - Red Hat Ansible Lightspeed を使用するには、**Ansible > Lightspeed: Enabled**ボックスをオンにし、Lightspeed の URL を入力します。

設定については、VisualStudio マーケットプレイスのドキュメントの Ansible VS Code Extension by Red Hat ページに記載されています。

3.5. VS CODE 内のコンテナーに ANSIBLE 開発ツールをインストールする

Dev Containers VS Code エクステンションには、開発コンテナーの設定を保存するための設定ファイルが必要です。開発コンテナーの設定ファイルを作成した後に、VS Code のコンテナー内のディレクトリーを再度開きます。

前提条件

1. Podman、Podman Desktop、Docker、Docker Desktop などのコンテナー化プラットフォームをインストール済みである。
2. Red Hat ログインがあり、registry.redhat.io の Red Hat レジストリーにログインできる。
3. VS Code がインストールされていること。
4. VS Code に Ansible エクステンションをインストール済みである。
5. VS Code に Microsoft Dev Containers エクステンションをインストール済みである。

手順

1. VS Code で、開発コンテナーの設定ファイルを保存するディレクトリーを開きます。
2. **.devcontainer** というサブディレクトリーを作成します。
3. **.devcontainer** ディレクトリーに、**devcontainer.json** というファイルを作成します。Podman と Docker のどちらを使用しているかに応じて、異なる設定を使用する必要があります。

   - Podman または Podman Desktop を使用している場合は、**devcontainer.json** に次のテキストを追加します。

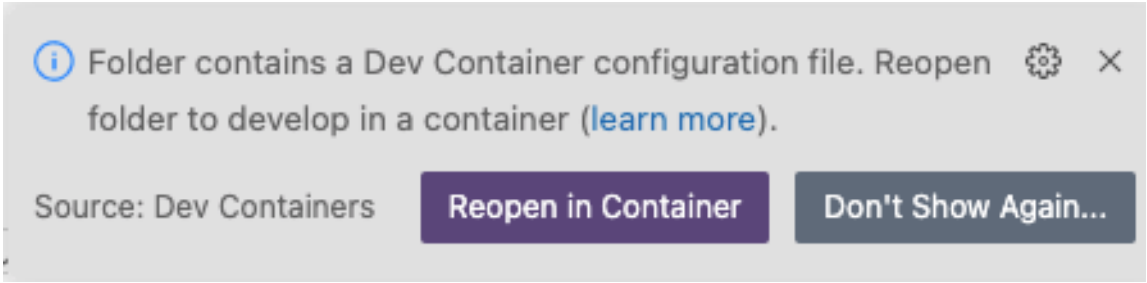
```
{
  "name": "ansible-dev-container-podman",
  "image": "registry.redhat.io/ansible-automation-platform-25/ansible-dev-tools-rhel8:latest",
  "containerUser": "root",
  "runArgs": [
    "--cap-add=SYS_ADMIN",
    "--cap-add=SYS_RESOURCE",
    "--device",
    "/dev/fuse",
    "--security-opt",
    "seccomp=unconfined",
    "--security-opt",
    "label=disable",
    "--security-opt",
    "apparmor=unconfined",
    "--userns=host",
    "--hostname=ansible-dev-container",
    "--volume",
    "ansible-dev-tools-container-storage:/var/lib/containers"
  ],
  "customizations": {
    "VS Codevscode": {
      "extensions": ["redhat.ansible"]
    }
  }
}
```

   - Docker または Docker Desktop を使用している場合は、**devcontainer.json** に次のテキストを追加します。

```
{
  "name": "ansible-dev-container-docker",
  "image": "registry.redhat.io/ansible-automation-platform-25/ansible-dev-tools-
rhel8:latest",
  "containerUser": "podman",
  "runArgs": [
    "--security-opt",
    "seccomp=unconfined",
    "--security-opt",
    "label=disable",
    "--cap-add=SYS_ADMIN",
    "--cap-add=SYS_RESOURCE",
    "--device",
    "/dev/fuse",
    "--security-opt",
    "apparmor=unconfined",
    "--hostname=ansible-dev-container"
  ],
  "updateRemoteUserUID": true,
  "customizations": {
    "VS Codevscode": {
      "extensions": ["redhat.ansible"]
    }
  }
}
```

4. コンテナー内のディレクトリーを再度開きます。

- ディレクトリーに **devcontainer.json** ファイルが含まれていることを VS Code が検出すると、次の通知が表示されます。



Reopen in Container をクリックします。

- 通知が表示されない場合は、**Remote** (><) アイコンをクリックします。表示されるドロップダウンメニューで、**Reopen in Container** を選択します。

VS Code ステータスバーの **Remote()** ステータスに **opening Remote** が表示され、通知にコンテナーを開く操作の進行状況が示されます。

検証

コンテナー内でディレクトリーが再度開かれると、**Remote()** ステータスに **Dev Container: ansible-dev-container** と表示されます。

3.6. RHEL 上のパッケージから ANSIBLE 開発ツールをインストールする

Ansible 開発ツールは、Ansible Automation Platform RPM (Red Hat Package Manager) パッケージにバンドルされています。Ansible Automation Platform のインストールの詳細は、**Red Hat Ansible Automation Platform インストールガイド** のドキュメントを参照してください。

前提条件

- RHEL がインストール済みである。
- Red Hat Subscription Manager でシステムを登録している。

手順

1. 次のコマンドを実行して、Simple Content Access (SCA) が有効化されているか確認します。

```
$ su subscription-manager status
```

Simple Content Access が有効化されている場合、出力には次のメッセージが含まれます。

```
Content Access Mode is set to Simple Content Access.
```

   a. Simple Content Access が有効化されていない場合は、Red Hat Ansible Automation Platform SKU をアタッチします。

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

2. 次のコマンドで Ansible 開発者ツールをインストールします。

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

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

検証:

Ansible 開発者ツールコンポーネントがインストールされていることを確認します。

```
$ rpm -aq | grep ansible
```

出力には、インストールされている Ansible パッケージが表示されます。

```
ansible-sign-0.1.1-2.el9ap.noarch
ansible-creator-24.4.1-1.el9ap.noarch
python3.11-ansible-runner-2.4.0-0.1.20240412.git764790f.el9ap.noarch
ansible-runner-2.4.0-0.1.20240412.git764790f.el9ap.noarch
ansible-builder-3.1.0-0.2.20240413.git167ed5c.el9ap.noarch
ansible-dev-environment-24.1.0-2.el9ap.noarch
ansible-core-2.16.6-0.1.20240413.gite636132.el9ap.noarch
python3.11-ansible-compat-4.1.11-2.el9ap.noarch
python3.11-pytest-ansible-24.1.2-1.el9ap.noarch
ansible-lint-6.14.3-4.el9ap.noarch
```

```
ansible-navigator-3.4.1-2.el9ap.noarch
python3.11-tox-ansible-24.2.0-1.el9ap.noarch
ansible-dev-tools-2.5-2.el9ap.noarch
```

インストールが成功すると、ansible-creator のヘルプドキュメントを表示できます。

```
$ ansible-creator --help

usage: ansible-creator [-h] [--version] command ...

The fastest way to generate all your ansible content.

Positional arguments:
 command
  add           Add resources to an existing Ansible project.
  init          Initialize a new Ansible project.

Options:
 --version      Print ansible-creator version and exit.
 -h     --help  Show this help message and exit
```

第4章 PLAYBOOK プロジェクトの作成

4.1. PLAYBOOK プロジェクトのスキャフォールディング

次の手順では、Ansible VS Code エクステンションを使用して新しい Playbook プロジェクトをスキャフォールディングするプロセスを説明します。

1. 前提条件
   - Ansible 開発者ツールがインストール済みである。
   - Ansible VS Code エクステンションがインストール済みである。
   - プロジェクトを保存するディレクトリーを特定している。

手順

1. VS Code を開きます。
2. VS Code アクティビティーバーの Ansible アイコンをクリックして、Ansible エクステンションを開きます。
3. **Ansible content creator** セクションで **Get started** を選択します。
   Ansible content creator タブが開きます。
4. **Create** セクションで、**Ansible playbook project** をクリックします。
   Create Ansible Project タブが開きます。
5. **Create Ansible project** タブのフォームに、以下を入力します。
   - **Destination directory**: 新しい Playbook プロジェクトをスキャフォールディングするディレクトリーへのパスを入力します。

     注記

     既存のディレクトリー名を入力すると、スキャフォールディングプロセスによってそのディレクトリーの内容が上書きされます。スキャフォールディングプロセスでは、**Force** オプションを有効にした場合にのみ、既存のディレクトリーを使用できます。

     - Ansible 開発ツールのコンテナー化されたバージョンを使用している場合、宛先ディレクトリーパスはコンテナーに対する相対パスであり、ローカルシステム内のパスではありません。コンテナー内の現在のディレクトリー名を確認するには、VS Code のターミナルで **pwd** コマンドを実行します。コンテナー内の現在のディレクトリーが **workspaces** の場合は、**workspaces/<destination_directory_name>** と入力します。
     - ローカルにインストールされたバージョンの Ansible 開発ツールを使用している場合は、ディレクトリーへのフルパスを入力します (例: **/user/<username>/projects/<destination_directory_name>)**。
   - **SCM organization and SCM project** Playbook 用に作成したロールを保存できるディレクトリーとサブディレクトリーの名前を入力します。
6. 新しい Playbook プロジェクトをスキャフォールディングするディレクトリーの名前を入力します。

検証

プロジェクトディレクトリーが作成されると、**Create Ansible Project** タブの **Logs** ペインに、次のメッセージが表示されます。この例では、宛先ディレクトリー名は **destination_directory_name** です。

```
------------------- ansible-creator logs -------------------
    Note: ansible project created at /Users/username/test_project
```

プロジェクトディレクトリーに、次のディレクトリーとファイルが作成されます。

```
$ tree -a -L 5 .
├── .devcontainer
│   ├── devcontainer.json
│   ├── docker
│   │   └── devcontainer.json
│   └── podman
│       └── devcontainer.json
├── .gitignore
├── README.md
├── ansible-navigator.yml
├── ansible.cfg
├── collections
│   ├── ansible_collections
│   │   └── scm_organization_name
│   │       └── scm_project_name
│   └── requirements.yml
├── devfile.yaml
├── inventory
│   ├── group_vars
│   │   ├── all.yml
│   │   └── web_servers.yml
│   ├── host_vars
│   │   ├── server1.yml
│   │   ├── server2.yml
│   │   ├── server3.yml
│   │   ├── switch1.yml
│   │   └── switch2.yml
│   └── hosts.yml
├── linux_playbook.yml
├── network_playbook.yml
└── site.yml
```

第5章 ANSIBLE 開発者ツールで PLAYBOOK を作成して実行する

5.1. 最初の PLAYBOOK の作成

以下の手順では、VS Code で最初の Playbook を作成して実行する際に、Ansible 開発者ツールがどのように役立つかを説明します。

前提条件

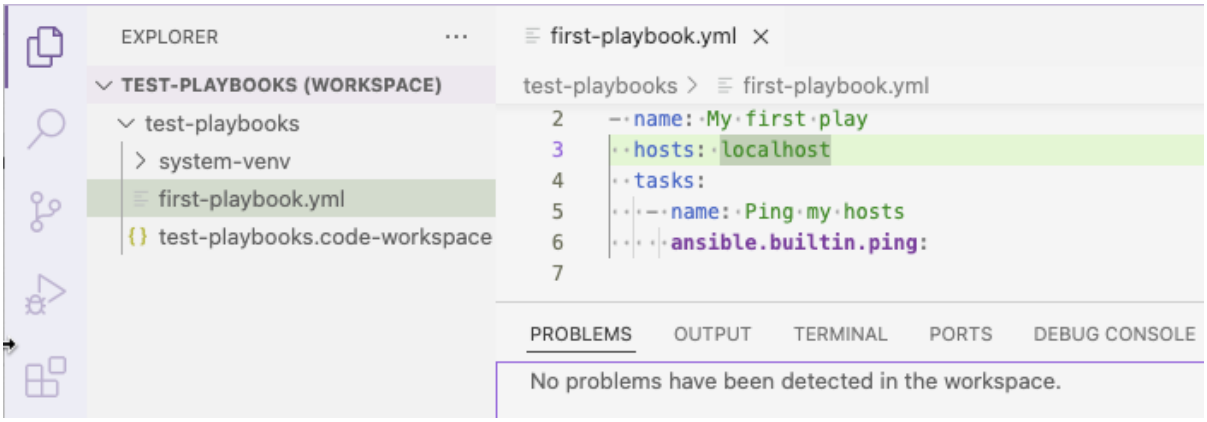
- Ansible VS Code エクステンションがインストール済みである。
- VS Code でターミナルを開いた。
- **ansible-devtools** がインストール済みである。

手順

1. VS Code で Playbook 用の新しい .yml ファイル (例: **example_playbook.yml)** を作成します。それを、サンプルの **site.yml** ファイルと同じディレクトリーレベルに配置します。
2. 次のサンプルコードを Playbook ファイルに追加し、ファイルを保存します。Playbook は、ローカルマシンへの **ping** を実行する単一のプレイで構成されます。

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

Ansible-lint は、バックグラウンドで実行され、ターミナルの **Problems** タブにエラーが表示されます。この Playbook にはエラーはありません。



3. Playbook ファイルを保存します。

5.2. PLAYBOOK の検査

Ansible VS Code エクステンションは、構文を強調表示して **.yml** ファイル内のインデントを支援します。

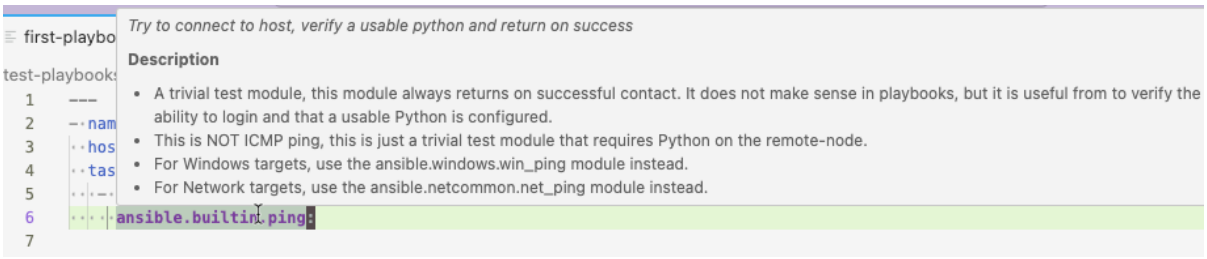
Playbook ファイルには次のルールが適用されます。

- すべての Playbook ファイルは、空白行で終わる必要があります。
- 行末の末尾にスペースを入れることはできません。
- すべての Playbook とすべてのプレイには識別子 (名前) が必要です。

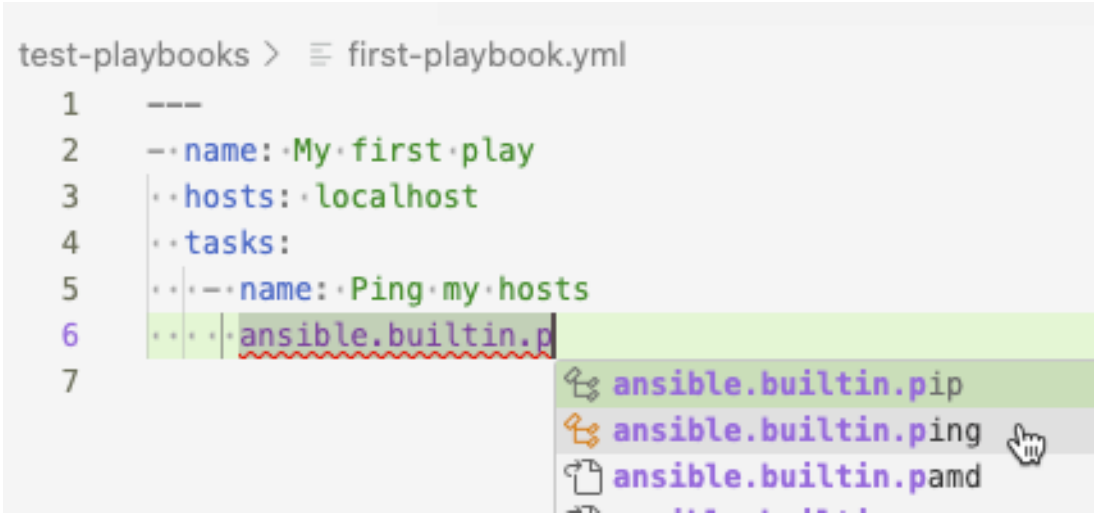
5.2.1. インラインヘルプ

Ansible エクステンションは、Playbook ファイルの編集時にインラインヘルプも提供します。

- キーワードまたはモジュール名の上にマウスをかざすと、Ansible エクステンションによってドキュメントが表示されます。



- モジュールの名前 (**ansible.builtin.ping** など) を入力し始めると、エクステンションによって候補のリストが表示されます。
  いずれかの候補を選択して行をオートコンプリートします。



5.3. PLAYBOOK を実行する

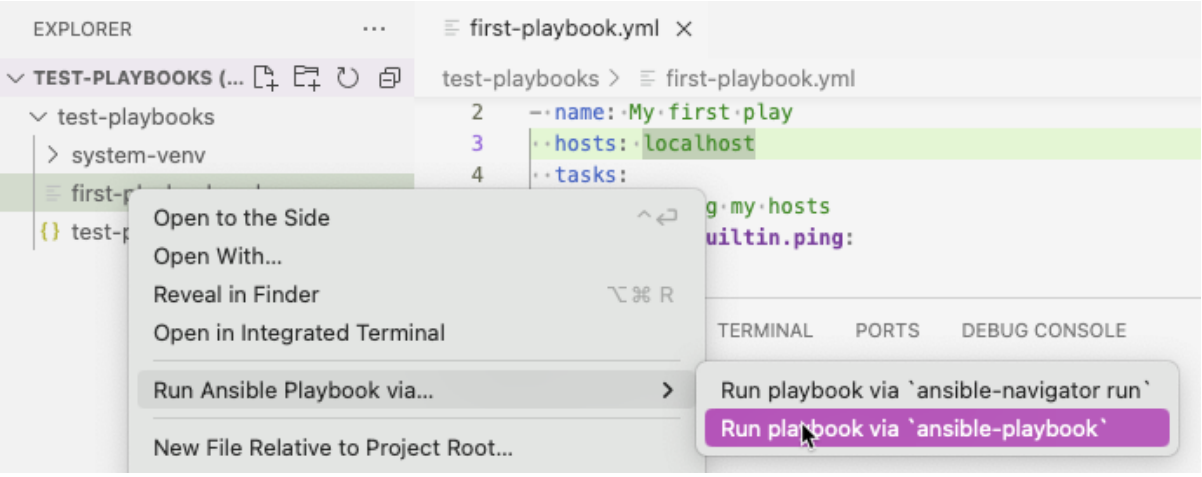
Ansible VS Code エクステンションには、Playbook を実行するための 2 つのオプションが用意されています。

- **ansible-playbook** は、Ansible Core を使用してローカルマシン上で Playbook を実行します。
- **ansible-navigator** は、Ansible Automation Platform が自動化ジョブを実行するのと同じ方法で、Playbook を実行環境で実行します。Ansible エクステンションの設定で、実行環境のベースイメージを指定します。

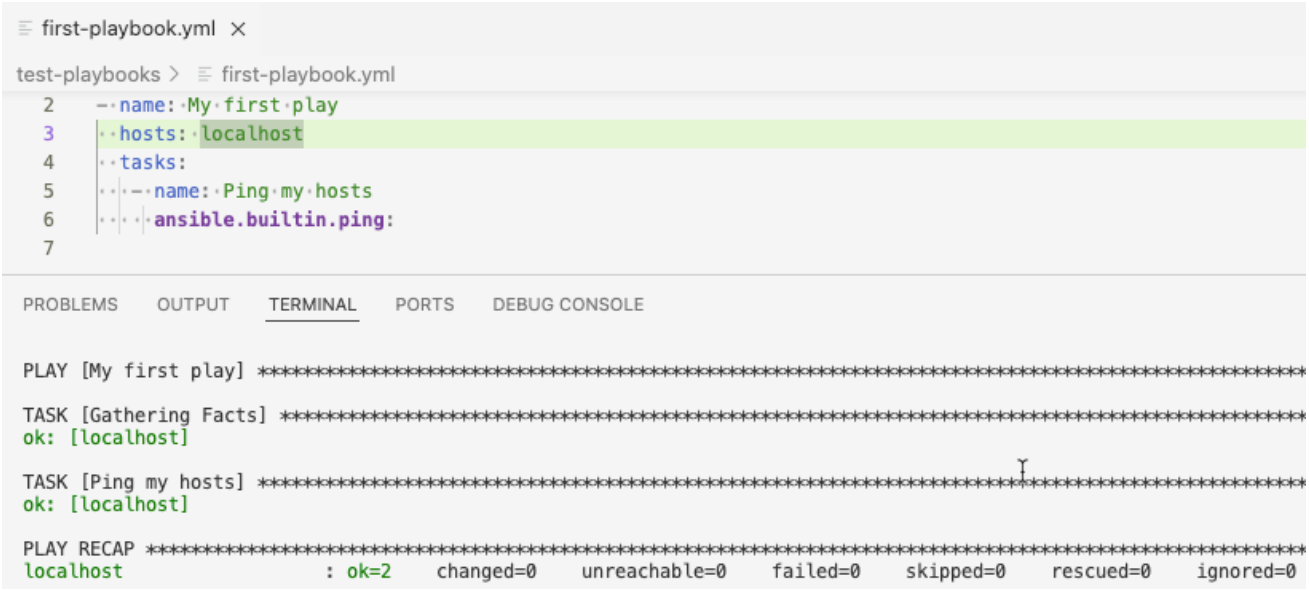
5.3.1. ansible-playbook で Playbook を実行する

手順

- Playbook を実行するには、**Explorer** ペインで Playbook 名を右クリックしてから、**Run Ansible Playbook via → Run playbook via ansible-playbook** を選択します。



出力は、VS Code ターミナルの **Terminal** タブに表示されます。**ok=2** および **failed=0** メッセージは、Playbook が正常に実行されたことを示します。



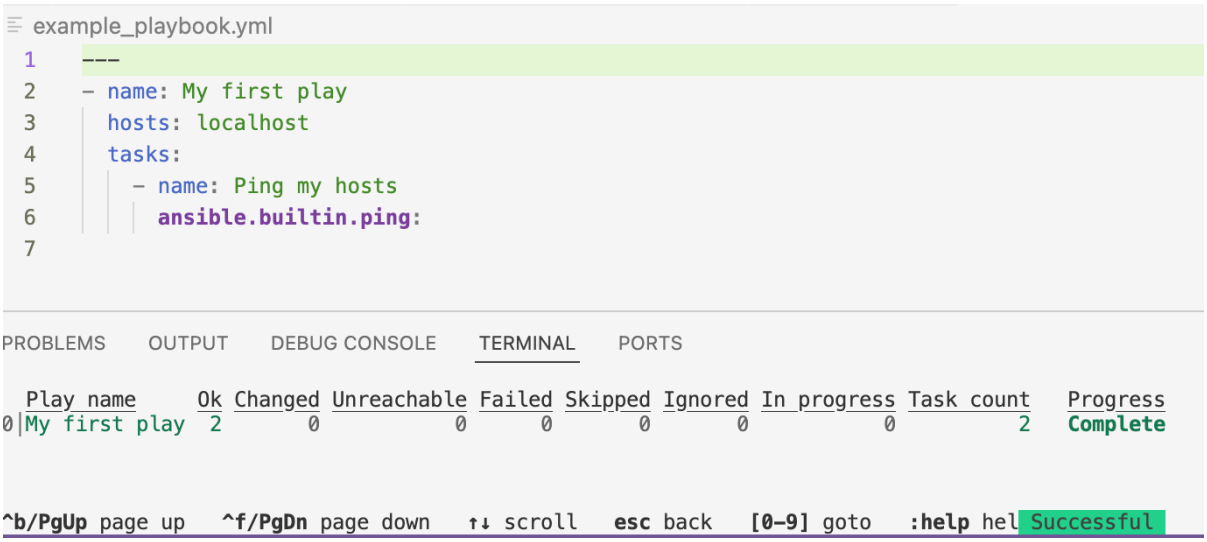
5.3.2. ansible-navigator で Playbook を実行する

前提条件

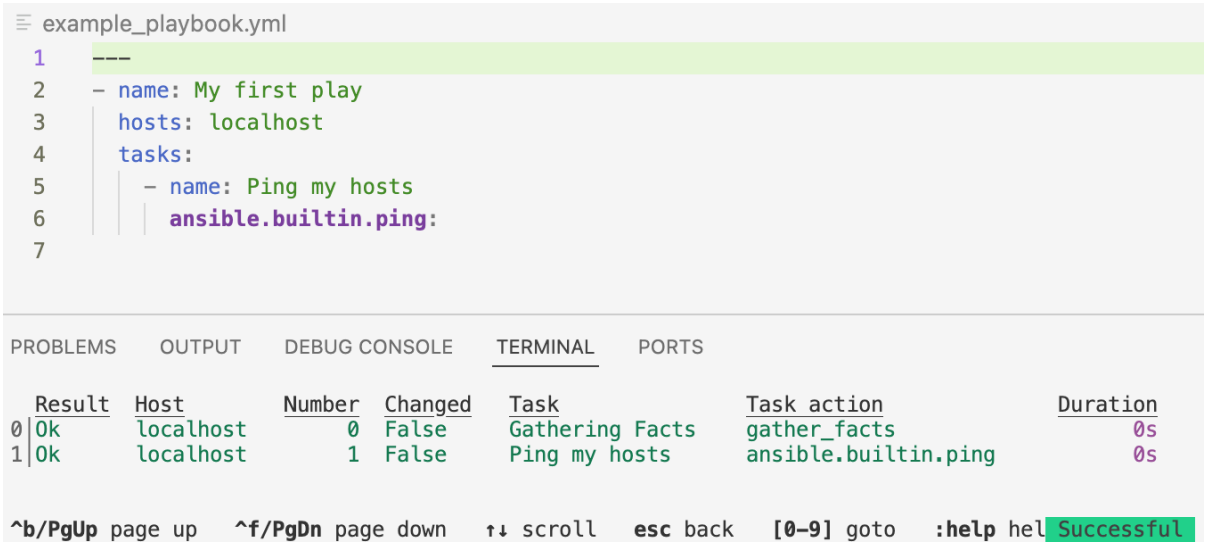
- Ansible エクステンション設定の **Ansible Execution Environment > Enabled**で、実行環境の使用を有効化した。
- **Ansible > Execution Environment: Image** に実行環境イメージのパスまたは URL を入力した。

手順

1. Playbook を実行するには、Explorer ペインで Playbook 名を右クリックしてから、**Run Ansible Playbook via → Run playbook via ansible-navigator run**を選択します。
出力は、VS Code ターミナルの **Terminal** タブに表示されます。**Successful** ステータスは、Playbook が正常に実行されたことを示します。



2. プレイの横にある番号を入力してプレイの結果を表示します。サンプル Playbook にはプレイが 1 つだけ含まれています。プレイで実行されたタスクのステータスを表示するには、**0** を入力します。



タスクの結果を確認するには、タスクの横の番号を入力します。

Automation content navigator を使用して Playbook を実行する方法の詳細は、Automation content navigator からの Playbook の実行 を参照してください。

5.3.3. 実行環境の操作

Red Hat が提供する自動化実行環境は、Red Hat エコシステムカタログ で確認できます。

ダウンロード方法を表示するには、実行環境をクリックします。

たとえば、最小 RHEL 8 ベースイメージをダウンロードするには、次のコマンドを実行します。

```
$ podman pull registry.redhat.io/ansible-automation-platform-25/ee-minimal-rhel9
```

ansible-builder を使用して、カスタム実行環境をビルドおよび作成できます。実行環境をローカルで操作する方法の詳細は、**実行環境の作成と使用** を参照してください。

実行環境をカスタマイズした後に、新しいイメージを Automation Hub のコンテナーレジストリーにプッシュできます。**実行環境の作成と使用** ドキュメントの 自動化実行環境の公開 を参照してください。実行環境の作成および消費

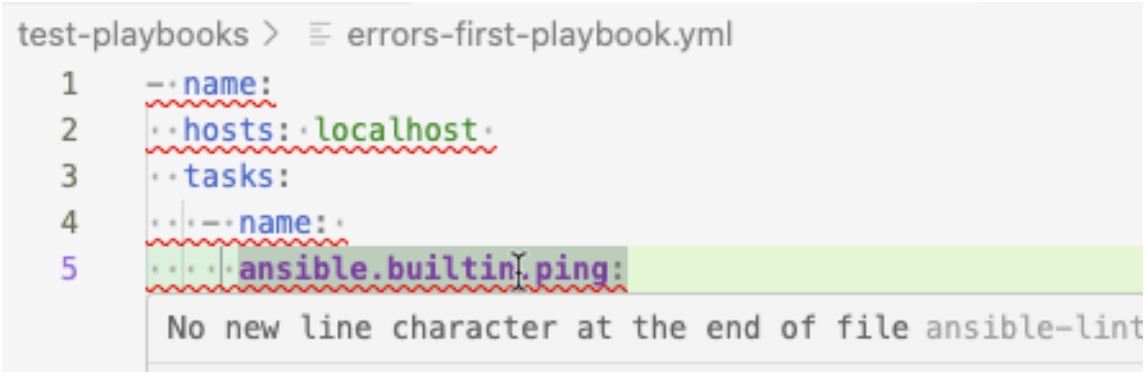
5.4. PLAYBOOK のデバッグ

5.4.1. エラーメッセージ

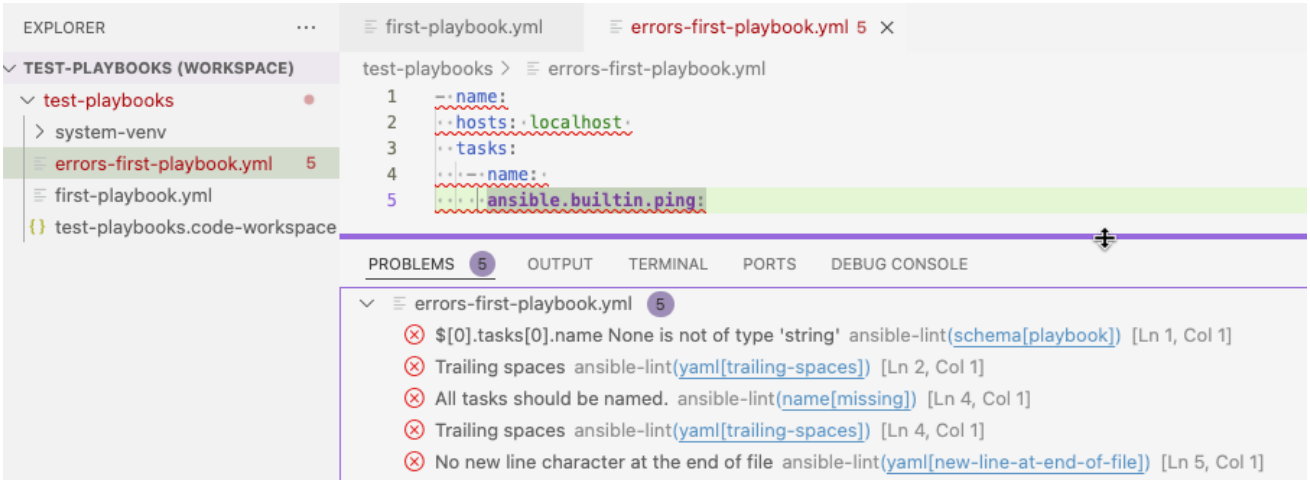
次の Playbook には複数のエラーが含まれています。

```
- name:
  hosts: localhost
  tasks:
   - name:
     ansible.builtin.ping:
```

エラーは VS Code で波線の下線で示されます。エラーの上にマウスをかざすと、詳細が表示されます。



エラーは、VS Code ターミナルの **Problems** タブに表示されます。エラーを含む Playbook ファイルは、**Explorer** ペインに番号で示されます。



$[0].tasks[0].name None is not of type 'string' は、Playbook にラベルがないことを示します。

5.5. PLAYBOOK のテスト

プロジェクトで Playbook をテストするには、ラボセットアップや仮想マシンなどの非実稼働環境で Playbook を実行します。

Automation content navigator (**ansible-navigator**) は、実行環境で Ansible コンテンツを開発およびトラブルシューティングするためのテキストベースのユーザーインターフェイス (TUI) です。

ansible-navigator を使用して Playbook を実行すると詳細な出力が生成され、それを検査することで Playbook が期待どおりに実行されているか確認できます。Playbook を実行する実行環境を指定すると、Ansible Automation Platform の実稼働環境セットアップをテストで複製できます。

- 実行環境で Playbook を実行するには、VS Code のターミナルから次のコマンドを実行します。

```
$ ansible-navigator run <playbook_name.yml> -eei <execution_environment_name>
```

たとえば、Ansible Automation Platform RHEL 9 の最小実行環境で **site.yml** という Playbook を実行するには、VS Code のターミナルから次のコマンドを実行します。

```
ansible-navigator run site.yml --eei ee-minimal-rhel8
```

出力はターミナルに表示されます。結果を検査し、実行された各プレイとタスクの詳細を確認できます。

Playbook の実行の詳細は **Automation content navigator creator ガイド**の Automation content navigator を使用した Ansible Playbook の実行 を参照してください。

第6章 ANSIBLE AUTOMATION PLATFORM での PLAYBOOK の公開と実行

次の手順では、新しい Playbook を Ansible Automation Platform のインスタンスにデプロイし、それを使用して自動化ジョブを実行する方法を説明します。

6.1. SCM にプロジェクトを保存する

Playbook プロジェクトを、GitHub などのソースコントロール管理システムのリポジトリーとして保存します。

手順

1. プロジェクトディレクトリーを Git リポジトリーとして初期化します。
2. プロジェクトを GitHub などのソース管理システムにプッシュします。

6.2. ANSIBLE AUTOMATION PLATFORM で PLAYBOOK を実行する

Ansible Automation Platform で Playbook を実行するには、Playbook プロジェクトを保存したリポジトリーの Automation Controller にプロジェクトを作成する必要があります。その後、プロジェクトから各 Playbook のジョブテンプレートを作成できます。

手順

1. ブラウザーで、Automation Controller にログインします。
2. 必要に応じて、ソース管理システムの Source Control 認証情報タイプを設定します。詳細は、**Automation Controller ユーザーガイド** の 新しい認証情報の作成 セクションを参照してください。
3. Automation Controller で、Playbook プロジェクトを保存した GitHub リポジトリーのプロジェクトを作成します。**Automation Controller ユーザーガイド** の プロジェクト の章を参照してください。
4. 作成したプロジェクトの Playbook を使用するジョブテンプレートを作成します。**Automation Controller ユーザーガイド** の ジョブテンプレート の章を参照してください。
5. ジョブテンプレートを起動することで、Automation Controller から Playbook を実行します。**Automation Controller ユーザーガイド** の ジョブテンプレートの起動 セクションを参照してください。