Learning Objectives

Learners will be able to...

- **Discover Ansible Galaxy**
- **Make use of third party modules**
- **Utilize search functionality in Ansible Galaxy**

Ansible Galaxy

Ansible Community

Ansible ships with hundreds of modules to manage infrastructure, but many users turn to the collections available from Ansible Community to extend Ansible's functionality.

Collections, the format of distributed content from Ansible Community, contain modules, plugins, roles, and even entire playbooks for developers to utilize. Ansible maintains an index of the latest versions of the most commonly used collections in the Collection Index, found here.

Ansible Galaxy

To get access to even more roles to implement in your playbooks, Ansible Galaxy can be utilized to discover third party (not affiliated with Ansible) modules. Galaxy is simply a repository that contains pre-written items to be used in a playbook. Ansible's built-in client `ansible-galaxy` can be used to download roles directly from Ansible Galaxy (the same client is also used for creating and uploading roles you create).

For example, if you want to install and configure MongoDB for creating replica data sets, search the following command in the terminal here.

```
ansible-galaxy search mongodb
```

The Ansible Galaxy website, https://galaxy.ansible.com/, can also be used to find community created content that includes user scores, number of downloads, and links to the source repos. From the Ansible Galaxy homepage, select "Database" repo and search MongoDB.

Installing Collection

How to Install a Collection

Once you have found the collection you want to install, it can be installed locally on the control node using
`ansible-galaxy collection install <collection>`

For our example we'll be installing the `community.MongoDB` collection, which can be found in Collections in the Community Namespace here. It can be installed using the following command:

```
ansible-galaxy collection install community.mongodb
```

By default it will be installed into your `~/.ansible/collections` folder, where it will be added to the stack. This is the default because in general it's not good practice to include code from a third party in your repository.

Having said this, it is possible to specify different directory like `./collections` to keep it with your repo using `-p`.

```
ansible-galaxy collection install community.mongodb -p
./collections
```

After running the above command, you should now see the collections folder in your file tree.

Depending on your needs, you can keep third party collections and roles within your repository or install them before applying the role.

Versioning

Specifying Versions

Since Ansible Galaxy's repository is public and its contents evolve over time, you may want to specify a version of a collection to avoid future compatibility issues.

We can specify the required version of a collection using

`ansible-galaxy collection install <collection-name>:==<version-number>`

As an example, to install version 1.4.2 of MongoDB we would run the following command:

```
ansible-galaxy collection install community.mongodb:==1.4.2
```

This is particularly useful if a new collection version does not support the version of software you want or changes in parameters made it incompatible with your use-case.

Specifying Version Ranges

Should you want to automatically follow patch updates, it is possible to specify range of versions. For example, to specify a range that is greater than or equal to 1.4.0, but less than 2.0.0, you would use the following command:

```
ansible-galaxy collection install community.mongodb:">=1.4.0,
<2.0.0"
```

The following commands are used to specify the range:

- `*`: The most recent version. This is the default.
- `!=`: Not equal to the version specified.
- `==`: Exactly the version specified.
- `>=`: Greater than or equal to the version specified.
- `>`: Greater than the version specified.

- <=: Less than or equal to the version specified.
- <: Less than the version specified.

Requirements

Defining System Dependencies

As discussed previously, our playbooks and roles can depend on other, external roles and collections to execute successfully. We call these **dependencies** and they can be defined in the `meta/main.yml` file of a role or in the `requirements.yml` file of a playbook.

On the previous page we used commands to install individual collections hosted in Galaxy. To define multiple dependencies to automatically be installed before executing your playbook, `requirements.yml` should be used.

> info
>
> The `requirements.yml` file should be placed in the same directory as the playbook or role that it is associated with.

Here's an example of a `requirements.yml` file that specifies two collections (`community.general` and `ansible.posix`) and one role (`geerlingguy.java`) that are required to execute a playbook or role:

```
---
collections:
  - name: community.general
    version: 3.6.0
  - name: ansible.posix
    version: 1.2.0
roles:
  - name: geerlingguy.java
    version: 4.0.0
```

You can specify the following keys for each collection entry:

- `name` specifies a collection or role, and implicitly determines the source
  - The optional `type` key can be used to specify the source but is rarely needed or required
- `version` specifies the version of the collection or role that should be installed
- `signatures` can be used to ensure the dependency file contents have not been tampered with

- The not-often used `source` key specifies sources of dependencies other than the default Ansible Galaxy repository (this could your custom repo , one managed by an employer, etc.)
- `type`, also rarely used, can be set to `galaxy`, `git`, `url`, etc.

To install dependencies from `requirements.yml` (which only contains the two collections from the example above, `community.general` and `ansible.posix`, and no roles), we run the following command:

```
ansible-galaxy install -r requirements.yml
```

Alternatively we can use `ansible-galaxy collection install -r` to install only the collections or `ansible-galaxy role install -r` for roles.