Ansible Roles

AGENDA

- ▶ What is Role?
- ▶ Find Roles & Ansible Galaxy
- ▶ Use Roles

What is Ansible Role?

What is Ansible Role?

- Ansible **role** is an independent component which allows reuse of common configuration steps.
- It is used within playbook.

mysql
- Installing Pre-requisites
- Installing mysql packages
- Configuring mysql service
- Configuring database and users

ngnix
- Installing Pre-requisites
- Installing ngnix packages
- Configuring ngnix service
- Configuring custom web pages

ondia

5

What is Ansible Role?

playbook1.yml
```
- name: Install and Configure MySql
  hosts: db-serves
  tasks:
    - name: Install Pre-requisites
      yum: name=pre-req-packages state=present

    - name: Install MySql Packages
      yum: name= mysql state=present

    - name: Start MySql Service
      service: name=mysql state=started

    - name: Configure Database
      mysql_db: name=db1 state=present
```

playbook2.yml
```
- name: Install and Configure MySql
  hosts: db-server1...db-server100
  roles:
    - mysql
```

mysql
- Installing Pre-requisites
- Installing mysql packages
- Configuring mysql service
- Configuring database and users

MySql-Role
```
tasks:
  - name: Install Pre-requisites
    yum: name=pre-req-packages state=present

  - name: Install MySql Packages
    yum: name= mysql state=present

  - name: Start MySql Service
    service: name=mysql state=started

  - name: Configure Database
    mysql_db: name=db1 state=present
```

Re-use

ondia

6

What is Ansible Role?

ansible-galaxy init <role_name>

To create a role using the *ansible-galaxy* command, we can simply use the below syntax in our terminal:

```
ansible-galaxy init <ROLE_NAME>
```

Running this command will give us a basic directory structure in our current working directory and this directory structure will be like the one below:

```
roles-example/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

What is Ansible Role?

ansible-galaxy init <role_name>

Defaults:

The *defaults* directory is for defining the variable defaults. The variables in *default* have the lowest priority thus becoming easy to override. If definition of a variable is nowhere else, the variable in *defaults/main.yml* will be used.

Files:

We use the *files* directory to add files that are needed by provisioning machine, without modification. Mostly, we use copy task for referencing files in the *files* directory. The most interesting part about this is that Ansible does not require a path for resources stored in *files* directory when working in the role.
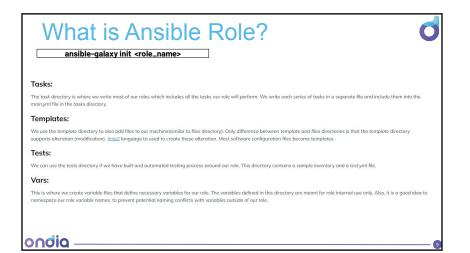
Handlers:

The *handlers* directory is used for storing Ansible handlers. Handlers are tasks that may be flagged during a play to run at the play's completion. We can have as many and as few handlers as we need.
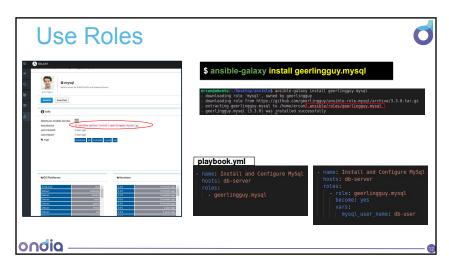
Meta:

We use the *meta* directory to store authorship information which is useful if we choose to publish our role on *galaxy.ansible.com*. The metadata of an Ansible role consists of author, supported platforms, and dependencies.

What is Ansible Role?

`ansible-galaxy init <role_name>`

Tasks:

The *task* directory is where we write most of our roles which includes all the tasks our role will perform. We write each series of tasks in a separate file and include them into the *main.yml* file in the *tasks* directory.

Templates:

We use the *template* directory to also add files to our machine(similar to *files* directory). Only difference between *template* and *files* directories is that the *template* directory supports alteration (modification). jinja2 language to used to create these alteration. Most software configuration files become templates.

Tests:

We can use the *tests* directory if we have built and automated testing process around our role. This directory contains a sample inventory and a *test.yml* file.

Vars:

This is where we create variable files that define necessary variables for our role. The variables defined in this directory are meant for role internal use only. Also, it is a good idea to namespace our role variable names, to prevent potential naming conflicts with variables outside of our role.

Find Roles & Ansible Galaxy

Use Roles


```
$ ansible-galaxy install geerlingguy.mysql
```

```
ercan@ubuntu:~/Desktop/ansible$ ansible-galaxy install geerlingguy.mysql
- downloading role 'mysql', owned by geerlingguy
- downloading role from https://github.com/geerlingguy/ansible-role-mysql/archive/3.3.0.tar.gz
- extracting geerlingguy.mysql to /home/ercan/.ansible/roles/geerlingguy.mysql
- geerlingguy.mysql (3.3.0) was installed successfully
```

playbook.yml

```
- name: Install and Configure MySql
  hosts: db-server
  roles:
    - geerlingguy.mysql
```

```
- name: Install and Configure MySql
  hosts: db-server
  roles:
    - role: geerlingguy.mysql
      become: yes
      vars:
        mysql_user_name: db-user
```

THANKS!

Any questions?

Powered by
Clarusway

Feature	Ansible Roles	Ansible Collections
Definition	A structured way to organize playbooks, tasks, handlers, variables, and templates.	A bundle of multiple roles, modules, plugins, and playbooks.
Scope	Focused on a specific function (e.g., configuring a web server).	A broader package that may include multiple roles and additional components.
Location	Stored inside `roles/` directory of a playbook.	Installed separately from Ansible Galaxy or custom sources.
Usage	Used within a playbook to organize tasks neatly.	Provides reusable automation content like roles, modules, and plugins.
Installation	Typically copied manually into `roles/` or downloaded via `ansible-galaxy install`.	Installed using `ansible-galaxy collection install <collection_name>`.
Content	Contains tasks, handlers, variables, and templates for a specific automation function.	Can include roles, custom modules, plugins, and documentation.
Example	A role named `apache` to install and configure Apache.	A collection named `community.general` providing various roles, modules, and plugins.

Difference Between Ansible Roles and Collections

Summary

- **Roles** help structure and reuse tasks in playbooks.
- **Collections** are a larger package that can contain **multiple roles, modules, and plugins** to extend Ansible's functionality.

Collections provide a **more modular and scalable** way to manage automation, while roles are a **smaller, focused** way to organize playbook tasks