Ansible Roles

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Session - 22 Agenda:
1. Ansible Roles

Ansible Roles:
Roles provide a framework for fully independent, or interdependent collections of variables, tasks, files, templates, and modules.

In Ansible, the role is the primary mechanism for breaking a playbook into multiple files. This simplifies writing complex playbooks, and it makes them easier to reuse. The breaking of playbook allows you to logically break the playbook into reusable components.

Each role is basically limited to a particular functionality or desired output, with all the necessary steps to provide that result either within that role itself or in other roles listed as dependencies.

Roles are not playbooks. Roles are small functionality which can be independently used but have to be used within playbooks. There is no way to directly execute a role. Roles have no explicit setting for which host the role will apply to.

Top-level playbooks are the bridge holding the hosts from your inventory file to roles that should be applied to those hosts.

Ansible roles make it easy to share the code with other users through Ansible Galaxy. You can upload the code in form of roles on Ansible Galaxy so that other users can download use it in their projects after making required changes.

What is an Ansible galaxy?
Ansible Galaxy is essentially a large public repository of Ansible roles. Roles ship with READMEs detailing the role's use and available variables. Galaxy contains a large number of roles that are constantly evolving and increasing. Galaxy can use git to add other role sources, such as GitHub.
https://galaxy.ansible.com/

Creating a New Role:
The directory structure for roles is essential to create a new role.

Role Structure:
Roles have a structured layout on the file system. The default structure can be changed but for now let us stick to defaults. Each role is a directory tree in itself. The role name is the directory name within the /roles directory.

Ansible galaxy command can be used to create the ansible roles directory tree.
$ ansible-galaxy -h

Usage:
ansible-galaxy [delete|import|info|init|install|list|login|remove|search|setup] [--help] [options] ...

Options:
-h, --help − Show this help message and exit.
-v, --verbose − Verbose mode (-vvv for more, -vvvv to enable connection debugging)
--version − Show program's version number and exit.

Ansible Roles

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Creating a Role Directory:

The above command has created the role directories.

$ ansible-galaxy init nehraclassesrole

You can create roles directories offline using --offline option.

$ ansible-galaxy init --force --offline nehraclassesrole

- nehraclassesrole was created successfully.

You can list the directory tree of the role directory created by you.

$ tree nehraclassesrole/

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

Directory Structure:

tasks - contains the main list of tasks to be executed by the role.
handlers - contains handlers, which may be used by this role or even anywhere outside this role.
defaults - default variables for the role. Defaults has the lower priority than vars.
vars - Other variables for the role. Vars has the higher priority than defaults.
files - contains files required to transfer or deployed to the target machines via this role.
templates - contains templates (jinja2) which can be deployed via this role.
meta - defines data/information about this role (author, dependency, versions, examples, etc.)
tests - Contains add-on files. This folder contains a test environment with an inventory file and a playbook script to test the role.

What are Ansible roles?

1. Ansible roles are consists of many playbooks, which is similar to modules in puppet and cook books in chef.
2. Roles are a way to group multiple tasks together into one container to do the automation in very effective manner with clean directory structures.
3. Ansible roles are set of tasks and additional files for a certain role which allow you to break up the configurations.
4. Ansible Roles allow reusing the codes by anyone if the role is suitable to him/her.
5. Ansible Roles can be easily modified and will reduce the syntax errors.

Ansible Roles

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Ansible Roles For Apache Web Server:

Let's understand roles using an example of Apache web server configuration using ansible jinja2 template.

$ vim index.j2

```
<html>
 <center><h1> The Apache webserver is running on {{ ansible_hostname }} </h1>
 </center>
</html>
```

Let's create an ansible playbook for the configuration of Apache web server on the managed nodes using above jinja2 template.

$ vim apache.yml

```
---
- name: Apache Web Server Configuration Playbook
  hosts: node1
  become: true
  tasks:
    - name: Installing Apache Latest Packages
      yum:
        name: httpd
        state: latest
    - name: Starting & Enabling Apache HTTPD Service
      service:
        name: httpd
        state: started
        enabled: true
    - name: Setting Up Apache Web Server
      template:
        src: /home/vikasnehra/index.j2
        dest: /var/www/html/index.html
    - name: Restarting Apache HTTPD Service
      service:
        name: httpd
        state: restarted
    - name: Allowing HTTP Traffic in the Firewall.
      firewalld:
        service: http
        permanent: yes
        state: enabled
    - name: Reload the Firewall Service
      service:
        name: firewalld
        state: reloaded
...
```

Execute the playbook.

$ ansible-playbook apache.yml

It may throw an error because of missing firewalld module, which we can install from ansible galaxy. Install the additional modules from the ansible galaxy using below command.

Ansible Roles

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

$ ansible-galaxy collection install ansible.posix

Now, we can execute the ansible playbook to install the Apache web server in the managed nodes.
$ ansible-playbook apache.yml

Verify the working of the Apache web server on node1, by using the curl command or using the web browser.
$ curl node1

Let's convert the above playbook into Ansible Roles for Apache. We will go to /etc/ansible directory and list the contents to see the roles directory there. This is default location for the ansible roles, however you can create roles anywhere in the system but in that case you have to mention its path in the ansible.cfg file.
$ cd /etc/ansible
$ ls -lh

Now, move to roles directory and list the contents. You will not find any roles there since you haven't created any roles yet.
$ cd roles/
$ ls -lh

First of all, we will create the directory structure for the ansible roles using the below command.
$ sudo ansible-galaxy init /etc/ansible/roles/apache --offline

You can list these directories using ls or tree command.
$ tree apache/

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

Let's go to the tasks directory first and edit the main.yml file there to mention the tasks.
$ cd /etc/ansible/roles/apache/tasks
$ sudo vim main.yml

Ansible Roles

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Here in this file, we can mention all the tasks that we want to push on the managed nodes but it is not a good practice. We are using ansible roles to reduce the complexity and making it easier to use. So, we can split the tasks into multiple files which will make the roles easily to understand and implement and at the same time it can be used in other automations jobs as well.

Let's create different files separately for each task under the tasks directory.

$ sudo vim install.yml

```
- name: Installing Apache Latest Packages
  yum:
     name: httpd
     state: latest
...
```

$ sudo vim service.yml

```
---
- name: Starting & Enabling Apache HTTPD Service
  service:
     name: httpd
     state: started
     enabled: true
...
```

$ sudo vim config.yml

```
---
 - name: Setting Up Apache Web Server
   template:
     src: templates/index.j2
     dest: /var/www/html/index.html
 - name: Allowing HTTP Traffic in the Firewall.
   firewalld:
     service: http
     permanent: yes
     state: enabled
 - name: Reload the Firewall Service
   service:
     name: firewalld
     state: reloaded
...
```

$ sudo vim main.yml

```
- import_tasks: install.yml
- import_tasks: service.yml
- import_tasks: config.yml
```

$ ls -lh

Now go to the templates directory to create the ansible jinja2 template files which will get copied to the managed nodes.

$ cd /etc/ansible/roles/apache/templates

$ sudo vim index.j2

Ansible Roles

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
<html>
 <center><h1> The Apache webserver is running on {{ ansible_hostname }} </h1>
 </center>
</html>
```

Now go to the handlers directory and edit the main.yml file to mention the httpd service restart task there.

```
$ cd /etc/ansible/roles/apache/handlers
$ sudo vim main.yml
- name: Restarting Apache HTTPD Service
  service:
     name: httpd
     state: restarted
```

Now go to the meta directory and edit the main.yml file to set the authorship information and other details.

```
$ cd /etc/ansible/roles/apache/meta
$ sudo vim main.yml
```

Now go to /etc/ansible directory and create a setup.yml file to implement the roles.

```
$ cd /etc/ansible
$ sudo vim setup.yml
---
- name: Apache Server Setup Playbook
  hosts: node3
  become: true
  roles:
     - apache
...
```

Now check the playbook and roles for the syntax errors. (Create some error in any file and verify the same.)

```
$ ansible-playbook setup.yml --syntax-check
```

Now, execute the playbook to implement the tasks using the roles to setup apache web server on node3.

```
$ ansible-playbook setup.yml
```

Verify the working of the Apache web server on node3, by using the curl command or using the web browser.

```
$ curl node3
```

Thanks