PLAYBOOKS

Install git in ami linux2

git_install.yml

```
---
- hosts: 172.31.16.196
  become: yes
  gather_facts: False
  tasks:
    - name: Installing git
      yum:
        name: git
        state: present
```

Install maven in ami linux2

[root@ip-172-31-18-141 ansible]# cat maven_install.yml

```
---
- hosts: 172.31.16.196
  become: yes
  tasks:
    - name: download maven and install
      shell: |
        wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo
        sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo
        yum install -y apache-maven
```

ansible galaxy

Galaxy is a hub for finding and sharing Ansible content.

Use Galaxy to jump-start your automation project with great content from the Ansible community. Galaxy provides pre-packaged units of work known to Ansible as roles.

Roles can be dropped into Ansible PlayBooks and immediately put to work. You'll find roles for provisioning infrastructure, deploying applications, and all of the tasks you do everyday.

Creating Roles

What is ansible roles ?

A role enables the sharing and reuse of Ansible tasks. It contains Ansible playbook tasks, plus all the supporting files, variables, templates, and handlers needed to run the tasks. A role is a complete unit of automation that can be reused and shared.

In practical terms, a role is a directory structure containing all the files, variables, handlers, Jinja templates, and tasks needed to automate a workflow. When a role is created, the default directory stucture contains the following:

```
[root@ip-172-31-18-141 app_deploy]# tree
.
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files
```

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.

vars - other variables for the role.

files - contains files which can be deployed via this role.

templates - contains templates which can be deployed via this role.

meta - defines some meta data for this role.

Refer bellow link for more info..

https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html?highlight=roles&extIdCarryOver=true&sc_cid=701f2000001OH7YAAW

How to create a role using ansible-galaxy ?

Using the **ansible-galaxy** command line tool that comes bundled with Ansible, you can create a role with the **init** command. For example, the following will create a role directory structure called **test-role-1** in the current working directory

$ ansible-galaxy init test-role-1

https://galaxy.ansible.com/docs/contributing/creating_role.html refer for more

Using Roles

The classic (original) way to use roles is via the **roles**: option for a given play:

```
---
- hosts: webservers
  roles:
    - common
    - webservers
```

This designates the following behaviors, for each role 'x':

- If roles/x/tasks/main.yml exists, tasks listed therein will be added to the play.
- If roles/x/handlers/main.yml exists, handlers listed therein will be added to the play.
- If roles/x/vars/main.yml exists, variables listed therein will be added to the play.
- If roles/x/defaults/main.yml exists, variables listed therein will be added to the play.
- If roles/x/meta/main.yml exists, any role dependencies listed therein will be added to the list of roles (1.3 and later).
- Any copy, script, template or include tasks (in the role) can reference files in roles/x/{files,templates,tasks}/ (dir depends on task) without having to path them relatively or absolutely.

When used in this manner, the order of execution for your playbook is as follows:

- Any **pre_tasks** defined in the play.
- Any **handlers** triggered so far will be run.
- Each role listed in **roles** will execute in turn. Any role dependencies defined in the roles **meta/main.yml** will be run first, subject to tag filtering and conditionals.
- Any **tasks** defined in the play.
- Any handlers triggered so far will be run.
- Any **post_tasks** defined in the play.
- Any handlers triggered so far will be run.

Using the following keywords, roles can be imported and executed in an Ansible playbook play:

roles

import_role

include_role

refere this note

https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html?highlight=roles&extIdCarryOver=true&sc_cid=701f2000001OH7YAAW

https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_includes.html

https://github.com/ansible/ansible-examples ansible p-laybook examples

install tomcat and deploy application:

https://github.com/ybmadhu/tomcat_latest.git

playbook keywords

https://docs.ansible.com/ansible/latest/reference_appendices/playbooks_keywords.html

https://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html

End to end deployment trough ansible (instaqll before git,maven,java,apache-tomcat)

github -→maven → apache-tomcat

[root@ip-172-31-18-141 ansible]# cat git_projects.yml

```
---
 - hosts: nodes
   become: yes
   tasks:
    - name: get the source code
      git:
       repo: 'https://github.com/ybmadhu/spring3-mvc-maven-xml-hello-world.git'
       dest: /home/mylogin/hello
      tags: git_download
    - name: packing the maven build
      shell: " mvn clean package "
      args:
       chdir: /home/mylogin/hello
    - name: copy artifacts to tomcat
      copy:
       src: /home/mylogin/hello/target/spring3-mvc-maven-xml-hello-world-1.2.war
       dest: /opt/apache-tomcat-8.5.40/webapps/
       remote_src: true
```

$ ansible-playbook git_projects.yml --private-key=tomcat.pem

Skip some tasks by using tags

$ansible-playbook git_projects.yml --private-key=tomcat.pem --skip-tags git_download

```
$ansible-playbook git_projects.yml --private-key=tomcat.pem --tags "configuration,packages"
```

https://docs.ansible.com/ansible/latest/user_guide/playbooks_tags.html