Red Hat Enterprise Linux Automation with Ansible

FEEDBACK TRANSLATIONS CERTIFICATE OF ATTENDANCE

Guided Exercise: Creating Roles ☆

In this exercise, you will create an Ansible role that uses variables, files, templates, tasks, and handlers to deploy a network service.

Outcomes

You should be able to create a role that uses variables and parameters.

The `myvhost` role installs and configures the Apache service on a host. A template named `vhost.conf.j2` is provided that will be used to generate `/etc/httpd/conf.d/vhost.conf`.

Log in to `workstation` as `student` using `student` as the password.

On `workstation`, run the `lab role-create start` command. This creates the working directory, `/home/student/role-create`, and populates it with an Ansible configuration file and host inventory.

```
[student@workstation ~]$ lab role-create start
```

Procedure 7.2. Instructions

1. Change to the `/home/student/role-create` working directory.

```
[student@workstation ~]$ cd ~/role-create
[student@workstation role-create]$
```

2. Create the directory structure for a role called `myvhost`. The role includes fixed files, templates, tasks, and handlers.

```
[student@workstation role-create]$ mkdir -v roles; cd roles
mkdir: created directory 'roles'
[student@workstation roles]$ ansible-galaxy init myvhost
- myvhost was created successfully
[student@workstation roles]$ rm -rvf myvhost/{defaults,vars,tests}
removed 'myvhost/defaults/main.yml'
removed directory: 'myvhost/defaults'
removed 'myvhost/vars/main.yml'
removed directory: 'myvhost/vars'
removed 'myvhost/tests/inventory'
removed 'myvhost/tests/test.yml'
removed directory: 'myvhost/tests'
[student@workstation roles]$ cd ..
[student@workstation role-create]$
```

3. Edit the `main.yml` file in the `tasks` subdirectory of the role. The role should perform the following tasks:

- The httpd package is installed
- The `httpd` service is started and enabled
- The web server configuration file is installed, using a template provided by the role

3.1. Edit the `roles/myvhost/tasks/main.yml` file. Include code to use the `yum` module to install the httpd package. The file contents should look like the following:

```
---
# tasks file for myvhost

- name: Ensure httpd is installed
  yum:
    name: httpd
    state: latest
```

3.2. Add additional code to the `roles/myvhost/tasks/main.yml` file to use the `service` module to start and enable the `httpd` service.

```
- name: Ensure httpd is started and enabled
  service:
    name: httpd
    state: started
    enabled: true
```

3.3. Add another stanza to use the `template` module to create `/etc/httpd/conf.d/vhost.conf` on the managed host. It should call a handler to restart the `httpd` daemon when this file is updated.

```
- name: vhost file is installed
  template:
    src: vhost.conf.j2
    dest: /etc/httpd/conf.d/vhost.conf
    owner: root
    group: root
    mode: 0644
  notify:
    - restart httpd
```

3.4. Save your changes and exit the `roles/myvhost/tasks/main.yml` file.

4. Create the handler for restarting the `httpd` service. Edit the `roles/myvhost/handlers/main.yml` file and include code to use the `service` module, then save and exit. The file contents should look like the following:

```
---
# handlers file for myvhost

- name: restart httpd
  service:
    name: httpd
    state: restarted
```

5. Move the `vhost.conf.j2` template from the project directory to the role's `templates` subdirectory.

```
[student@workstation role-create]$ mv -v vhost.conf.j2 roles/myvhost/templates/
renamed 'vhost.conf.j2' -> 'roles/myvhost/templates/vhost.conf.j2'
```

6. Create the HTML content to be served by the web server.

6.1. Create the `files/html/` directory to store the content in.

```
[student@workstation role-create]$ mkdir -pv files/html
mkdir: created directory 'files/html'
```

6.2. Create an `index.html` file below that directory with the contents: `simple index`.

```
[student@workstation role-create]$ echo \
> 'simple index' > files/html/index.html
```

7. Test the `myvhost` role to make sure it works properly.

7.1. Write a playbook that uses the role, called `use-vhost-role.yml`. Include a task to copy the HTML content from `files/html/`. Use the `copy` module and include a trailing slash after the source directory name. It should have the following content:

```
---
- name: Use myvhost role playbook
  hosts: webservers
  pre_tasks:
    - name: pre_tasks message
      debug:
        msg: 'Ensure web server configuration.'

  roles:
    - myvhost

  post_tasks:
    - name: HTML content is installed
      copy:
        src: files/html/
        dest: "/var/www/vhosts/{{ ansible_hostname }}"

    - name: post_tasks message
      debug:
        msg: 'Web server is configured.'
```

> **NOTE**
>
> The trailing slash causes the source directory and all of its contents to be copied to the managed host.

7.2. Before running the playbook, verify that its syntax is correct by running `ansible-playbook` with the `--syntax-check`. If it reports any errors, correct them before moving to the next step. You should see output similar to the following:

```
[student@workstation role-create]$ ansible-playbook use-vhost-role.yml \
> --syntax-check

playbook: use-vhost-role.yml
```

7.3. Run the playbook. Review the output to confirm that Ansible performed the actions on the web server, `servera`.

```
[student@workstation role-create]$ ansible-playbook use-vhost-role.yml

PLAY [Use myvhost role playbook] ***********************************************

TASK [Gathering Facts] *********************************************************
ok: [servera.lab.example.com]

TASK [pre_tasks message] *******************************************************
ok: [servera.lab.example.com] => {
    "msg": "Ensure web server configuration."
}

TASK [myvhost : Ensure httpd is installed] *************************************
changed: [servera.lab.example.com]

TASK [myvhost : Ensure httpd is started and enabled] ***************************
changed: [servera.lab.example.com]

TASK [myvhost : vhost file is installed] ***************************************
changed: [servera.lab.example.com]

RUNNING HANDLER [myvhost : restart httpd] **************************************
changed: [servera.lab.example.com]

TASK [HTML content is installed] ***********************************
changed: [servera.lab.example.com]

TASK [post_tasks message] ******************************************************
ok: [servera.lab.example.com] => {
    "msg": "Web server is configured."
}

PLAY RECAP *********************************************************************
servera.lab.example.com    : ok=8    changed=5    unreachable=0    failed=0
```

7.4. Run ad hoc commands to confirm that the role worked. The httpd package should be installed and the `httpd` service should be running.

```
[student@workstation role-create]$ ansible webservers -a \
> 'systemctl is-active httpd'
servera.lab.example.com | CHANGED | rc=0 >>
active

[student@workstation role-create]$ ansible webservers -a \
> 'systemctl is-enabled httpd'
servera.lab.example.com | CHANGED | rc=0 >>
enabled
```

7.5. The Apache configuration should be installed with template variables expanded.

```
[student@workstation role-create]$ ansible webservers -a \
> 'cat /etc/httpd/conf.d/vhost.conf'
servera.lab.example.com | CHANGED | rc=0 >>
# Ansible managed:

<VirtualHost *:80>
    ServerAdmin webmaster@servera.lab.example.com
    ServerName servera.lab.example.com
    ErrorLog logs/servera-error.log
    CustomLog logs/servera-common.log common
    DocumentRoot /var/www/vhosts/servera/

    <Directory /var/www/vhosts/servera/>
        Options +Indexes +FollowSymlinks +Includes
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

7.6. The HTML content should be found in a directory called `/var/www/vhosts/servera`. The `index.html` file should contain the string "simple index".

```
[student@workstation role-create]$ ansible webservers -a \
> 'cat /var/www/vhosts/servera/index.html'
servera.lab.example.com | CHANGED | rc=0 >>
simple index
```

7.7. Use the `uri` module in an ad hoc command to check that the web content is available locally. Set the `return_content` parameter to `true` to have the content of the server's response added to the output. The server content should be the string `simple index\n`.

```
[student@workstation role-create]$ ansible webservers -m uri \
> -a 'url=http://localhost return_content=true'
servera.lab.example.com | SUCCESS => {
    "accept_ranges": "bytes",
    "changed": false,
    "connection": "close",
    "content": "simple index\n",
...output omitted...
    "status": 200,
    "url": "http://localhost"
}
```

7.8. Confirm that the web server content is available to remote clients.

```
[student@workstation role-create]$ curl http://servera.lab.example.com
simple index
```

Finish

Run the `lab role-create finish` command to clean up the managed host.

```
[student@workstation ~]$ lab role-create finish
```

This concludes the guided exercise.

← PREVIOUS (/ROL/APP/COURSES/RH294-8.4/PAGES/CH07S05) → NEXT (/ROL/APP/COURSES/RH294-8.4/PAGES/CH07S07)

Privacy Policy (http://s.bl-1.com/h/cZrgWbQn?url=https://www.redhat.com/en/about/privacy-policy?extIdCarryOver=true&sc_cid=701f2000001D8QoAAK)

Red Hat Training Policies (http://s.bl-1.com/h/cZrb2DXG?url=https://www.redhat.com/en/about/red-hat-training-policies)

Terms of Use (https://www.redhat.com/en/about/terms-use)

All policies and guidelines (https://www.redhat.com/en/about/all-policies-guidelines)

Release Notes (https://learn.redhat.com/t5/Red-Hat-Learning-Subscription/Red-Hat-Learning-Subscription-Release-Notes/ba-p/22952)

Cookie Preferences