Complete Ansible Automation Training

Advance Ansible Automation Features

Roles

- Roles simplifies long playbooks by grouping tasks into smaller playbooks

OR

- The role are the way of breaking a playbook into multiple playbook files. This simplifies writing complex playbooks, and it makes them easier to reuse

- Writing ansible code to manage the same service for multiple environments creates more complexity and it becomes difficult to manage everything in one ansible playbook. Also sharing code among other teams become difficult. That is where Ansible Role helps solve these problems

- Roles are like templates that are most of the time static and can be called by the playbooks

- Roles allow the entire configuration to be grouped in:
  - Tasks
  - Modules
  - Variables
  - Handlers

Example

Roles

Task list for east-webservers

```
---
- name: Setup httpd webserver
  hosts: east-webservers
  tasks:
1 - name: Install httpd packages
    yum:
     name: httpd
     state: present

2 - name: Start httpd
    service:
     name: httpd
     state: started

3 - name: Open port http on firewall
    firewalld:
     service: http
     permanent: true
     state: enabled

4 - name: Restart firewalld
    service:
     name: firewalld
     state: reloaded
```

Task list for west-webservers

```
---
- name: Setup httpd webserver
  hosts: west-webservers
  tasks:
1 - name: Install httpd packages
    yum:
     name: httpd
     state: present

2 - name: Start httpd
    service:
     name: httpd
     state: started
```

vim byrole.yml

Combined tasks

```
---
- name: Setup httpd webserver
  hosts: east-webservers
  tasks:
1 - name: Install httpd packages
    yum:
     name: httpd
     state: present

2 - name: Start httpd
    service:
     name: httpd
     state: started

3 - name: Open port http on firewall
    firewalld:
     service: http
     permanent: true
     state: enabled

4 - name: Restart firewalld
    service:
     name: firewalld
     state: restarted

  hosts: west-webservers
  tasks:
5 - name: Install httpd packages
    yum:
     name: httpd
     state: present

6 - name: Start httpd
    service:
     name: httpd
     state: started
```

Roles

```
---
- name: Setup httpd webserver
  hosts: east-webservers
  tasks:
1 - name: Install httpd packages
    yum:
     name: httpd
     state: present

2 - name: Start httpd
    service:
     name: httpd
     state: started

3 - name: Open port http on firewall
    firewalld:
     service: http
     permanent: true
     state: enabled

4 - name: Restart firewalld
    service:
     name: firewalld
     state: restarted


  hosts: west-webservers
  tasks:
5 - name: Install httpd packages
    yum:
     name: httpd
     state: present

6 - name: Start httpd
    service:
     name: httpd
     state: started
```

```
---
- name: Setup full httpd webserver
  tasks:
  - name: Install httpd packages
    yum:
     name: httpd
     state: present

  - name: Start htt
    servic



  - n        ort 80 for http access
    f   walld:
     service: http
     permanent: true
     state: enabled

  - name: Restart firewalld
    service:
     name: firewalld
     state: restarted
```

Roles Playbook: fullinstall

```
---
- name: Setup basic httpd webser
  tasks:
  - name: Install a
    yum:



  - n            ttpd
    se     e:
     name: httpd
     state: started
```

Roles Playbook: basicinstall

```
---
- name: Full install
  hosts: east-webservers
  roles:
  - fullinstall

- name: Basic install
  hosts: west-webservers
  roles:
  - basicinstall
```

Very simplified version of playbook

Roles

- Please note roles can be grouped by type of servers, type of applications or organizational requirement

- To create roles

```
# Go to ControlNode
# cd /etc/ansible/roles
```

Make directory for each role

```
# e.g mkdir [rolenames]
# mkdir basicinstall
# mkdir fullinstall
```

You can also create roles based on the type of servers:

```
e.g. # mkdir webservers
# mkdir dbservers
# mkdir appservers
```

Create sub-directory **<u>tasks</u>** within each directory

```
# mkdir basicinstall/tasks
# mkdir fullinstall/tasks
```

Create yml files within these sub-directories

```
# touch basicinstall/tasks/main.yml
# touch fullinstall/tasks/main.yml
```

Roles

```
# vim fullinstall/tasks/main.yml
```

```
---
- name: Install httpd package
  yum:
   name: httpd
   state: present

- name: Start httpd
  service:
   name: httpd
   state: started

- name: Open port for http
  firewalld:
   service: http
   permanent: true
   state: enabled

- name: Restart firewalld
  service:
   name: firewalld
   state: reloaded
```

```
# vim basicinstall/tasks/main.yml
```

```
---
- name: Install httpd package
  yum:
   name: httpd
   state: present

  - name: Start httpd
    service:
     name: httpd
     state: started
```

```
# vim /etc/ansible/playbooks/byrole.yml
```

```
---
- name: Full install
  hosts: all
  roles:
  - fullinstall

- name: Basic install
  hosts: localhosts
  roles:
  - basicinstall
```

By: Imran Afzal
www.utclisolutions.com

Roles by Application

```
---
- name: Install packages
  hosts: all
  tasks:
  - name: Install Apache package
    yum:
     name: httpd
     state: present

  - name: Install Time package
    yum:
     name: ntpd or chrony
     state: present

  - name: Install DNS package
     yum:
     name: named
     state: present
```

1

2

3

- To create roles

```
# Go to ControlNode
# cd /etc/ansible/roles
```

Make directory for each role

```
# mkdir apache
# mkdir ntpd
# mkdir named
```

Create sub-directory <u>**tasks**</u> within each directory

```
# mkdir apache/tasks
# mkdir ntpd/tasks
# mkdir named/tasks
```

Create yml files within these sub-directories

```
# touch apache/tasks/main.yml
# touch ntpd/tasks/main.yml
# touch named/tasks/main.yml
```

```
---
- name: Install packages
  hosts: all
  roles:
    - apache
    - ntpd
    - named
```

✓

By: Imran Afzal
www.utclisolutions.com

Roles on Ansible Galaxy

- You can find a ton of resources on roles through Ansible galaxy
- You can download pre-defined or pre-written roles from the Ansible galaxy
- www.galaxy.ansible.com

Tags

- Tags are the reference or aliases to a task
- Instead of running an entire Ansible playbook, use tags to target a specific tasks you need to run

```
# vim httpbytags.yml
---
- name: Setup Apache server
  hosts: localhost
  tasks:
  - name: Install httpd
    yum:
     name: httpd
     state: present
    tags: i-httpd

  - name: Start httpd
    service:
     name: httpd
     state: started
    tags: s-httpd
```

```
# anisble-playbook httpbytags.yml -t i-httpd
```

```
# anisble-playbook httpbytags.yml -t s-httpd
```

- To list all tags in a playbook
  ```
  # anisble-playbook httpbytags.yml --list-tags
  ```
- To run a task using tag
  ```
  # anisble-playbook httpbytags.yml -t i-httpd
  ```
- To skip a task using tag
  ```
  # anisble-playbook httpbytags.yml --skip-tags i-httpd
  ```
- Wait a second…
- We can use "tasks option" to start a playbook at a specific task
  ```
  # anisble-playbook yamlfile.yml --start-at-task 'Task name'
  # anisble-playbook http.yml --start-at-task 'Intall httpd'
  ```

By: Imran Afzal
www.utclisolutions.com

Variables

- Variables are like containers that hold the defined value which can be used repetitively

IMPORTANT Things to Remember about Variables!

- Name can include letters, numbers and underscore
- Name should always start with a letter
- Cannot have a spaces, dots (.) or hypen (-) in variable name
- Variables can be defined inside of inventory files as well

Variables

Example

1

```
---
- name: Install some package
  hosts: all
  vars:
   sespackage: sesquipedalianism

  tasks:
  - name: Package install
    yum:
     name: "{{ sespackage }}"
     state: present

  - name: Start service
    service:
     name: "{{ sespackage }}"
     state: started
```

```
---
- name: Package installation
  hosts: all
  vars:
   pack: httpd

  tasks:
  - name: Install package
    yum:
     name: "{{ pack }}"
     state: present

  - name: Start service
    service:
     name: "{{ pack }}"
     state: started
```

Variables

Examples

2

```
---
- name: Copy file to remote clients
  hosts: all
  vars:
   srcfile: /home/iafzal/somefile
  tasks:

  - name: Copying file
    become: true
    copy:
     src: "{{ srcfile }}"
     dest: /tmp
     owner: iafzal
     group: iafzal
     mode: 0644
```

3

```
---
- name: Create a file
  hosts: localhost
  vars:
   file_name: kramer

  tasks:
  - name: Create file in /tmp
    file:
     state: touch
     path: /tmp"{{ file_name }}".txt
```

4

```
---
 - name: Print Hello world
   hosts: all
   vars:
     say: Hello World!

   tasks:
   - name: Ansible Variable Basic Usage
       debug:
       msg: "{{ say }}"
```

By: Imran Afzal
www.utclisolutions.com

Variables in Inventory File

Example

```
[webservers]
client1.xyz.com
client2.xyz.com

[abc:vars]
fooserver=foo.abc.example.com
ntpserver=ntp.abc.example.com
proxyserver=proxy.abc.example.com


server1 ansible_host=201.0.113.111
server2 ansible_host=201.0.113.112
server3 ansible_host=201.0.113.113
server4 ansible_host=abc.example.com
```