ANSIBLE

Ansible Terminology

- **Play**: a provisioning executed from start to finish is called a *play*
- **Playbook**: the entry point for Ansible provisionings, where the automation is defined through tasks using YAML format
- **Facts**: global variables containing information about the system, like network interfaces or operating system
- **Handlers**: used to trigger service status changes, like restarting or stopping a service

Playbook

- Playbooks are Ansible's configuration, deployment, and orchestration language.
- playbooks can be used to manage configurations and deployments to remote machines.
- Playbooks are designed to be human-readable and are developed in a basic text language.
- Each playbook is composed of one or more 'plays' in a list.

YAML

- The **YAML** stands for **Yet Another Markup Language** which includes human readable content and often used in configuration files

Following are the features of YAML:

- Compare to XML or JSON, YAML is less complex and provides same features.
- It provides configuration settings without need to learn complex code types such as CSS, JavaScript or PHP.

Basic Rules of YAML

- You must end the YAML files with **.yaml** (or) **.yml** extension.
- YAML must be case sensitive.
- YAML doesn't support use of tabs. Instead of tabs, it uses spaces which are supported universally.

Basic Data Types of YAML

- YAML supports some basic data types which can be used with programming languages such as:
- **Scalars**: strings or numbers.
- **Sequences**: arrays or lists.
- **Mappings**: hashes or dictionaries.

Example Yaml File

```
---
- Martin:
      name: Martin smith
      job: Developer
      skills:
          - python
          - perl
- John:
    name: John pal
    job:   Developer
    skills:
          - lisp
          - fortran
```

Playbook Basics

Hosts and Users : each play in a playbook, you get to choose which machines in your infrastructure to target and what remote user to complete the steps (called tasks)

Ex:

```
---
- hosts: webservers
  remote_user: root
```

Playbook Basics

Tasks list: Each play contains a list of tasks. Tasks are executed in order, one at a time, against all machines matched by the host pattern. the playbook will runs in top to bottom

Ex:

```
tasks:
    - name: make sure apache is running
      service: name=httpd state=started
```

#vi playbooks/pb1.yml

```
- hosts: all
  gather_facts: false
  remote_user: root
  tasks:
  - name: to copy file
    ansible.builtin.copy:
      src: /root/files/demo.txt
      dest: /opt/demo.txt

  - name: to display msg
    ansible.builtin.debug:
      msg: file copied!
```

to validate playbook:

```
# ansible-playbook playbooks/pb1.yml --syntax-check
```

to Execute playbook:

```
# ansible-playbook playbooks/pb1.yml
```

```
- hosts: web
  remote_user: root
  tasks:
  - name: to update repo
    ansible.builtin.raw:
      apt-get update

  - name: to install apache
    ansible.builtin.apt:
      name: apache2
      state: present

  - name: to deploy a file
    ansible.builtin.copy:
      src: /root/files/index.html
      dest: /var/www/html/index.html

  - name: to start service
    ansible.builtin.service:
      name: apache2
      state: started
```

Handlers: Running Operations On Change

```
- hosts: ubnt
  tasks:
   - name: install apache
     apt: name=apache2 state=latest
     notify:
       - start apache
   - name: install tomcat
     apt: name=tomcat7 state=latest
     notify:
       - start tomcat
  handlers:
   - name: start apache
     service: name=apache2 state=started
   - name: start tomcat
     service: name=tomcat7 state=started
```

When Condition

```
---
- hosts: all
  become: true
  vars:
  - pack1: mysql-server
  - pack2: mariadb-server
  tasks:
  - name: to install {{pack1}}
    ansible.builtin.apt:
      name: "{{pack1}}"
      update_cache: yes
      state: present
    notify: start mysql
    when: ansible_distribution=='Ubuntu'
    changed_when: true
```

When Condition

```
- name: to install {{pack2}}
  ansible.builtin.yum:
    name: "{{pack2}}"
    update_cache: yes
    state: present
  notify: start mariadb
  when: ansible_distribution=='Amazon'
  changed_when: true

handlers:
- name: start mysql
  ansible.builtin.service:
    name: mysql
    state: started

- name: start mariadb
  ansible.builtin.systemd:
    name: mariadb
    state: started
```

Ansible Error Handling

```
---
- hosts: all
  become: true
  gather_facts: true
  tasks:
    - name: to install net-too
      package: name=net-too state=present
      ignore_errors: yes

    - name: to install git-core
      package: name=git-core state=present
      ignore_errors: yes
```