Ansible Playbook Keywords – Definitions

name
hosts
become
vars
tasks
handlers
gather_facts
roles
tags
loops

1. name

A label or description for a play or a task.
Helps users understand what the play/task is doing.
Not mandatory, but always recommended for readability.
Example:

```
name: Install Apache server
```

2. hosts

Specifies which target machines the play will run on.
Takes inventory group name, hostname, or pattern.
Example:

```
hosts: webservers
```

3. become

Used for privilege escalation .
It allows Ansible to switch from a normal user to root (or another privileged user) to run tasks that need admin permission.
Usually set as yes or true.
Example:

```
become: yes
```

4. vars

vars means variables in Ansible used to Avoid hardcoding values
A section inside a playbook used to define custom variables.
Makes playbooks dynamic and reusable.
Example:

```
vars:
    package_name: httpd
```

Pass value during run time:

```
ansible-playbook site.yml -e "package_name=git"
```

5. tasks

A list of actions for Ansible to execute on the target hosts.
Each task calls a module (e.g., yum, apt, file, copy).

Example:

```
tasks:
  - name: Install Apache
    yum:
      name: httpd
      state: present
```

6. handlers

Special tasks that run only when notified.
Usually used for actions that should happen only when something changes, like restarting a service after updating a config file.
Handlers run at the end of a play (after all regular tasks have run).
Define a handler under the handlers: section.
A normal task “notifies” the handler if it changes something.
Handlers run once per play
If multiple tasks notify the same handler, it runs only once at the end.
Example:

```
tasks:
  - name: Copy config file
    copy:
      src: myapp.conf
      dest: /etc/myapp.conf
    notify: restart myapp

handlers:
  - name: restart myapp
    service:
      name: myapp
      state: restarted
```

7. gather_facts

gather_facts is a playbook keyword that tells Ansible whether to collect information about the target host before executing tasks.
It collects the information like (like OS type, IP, memory, etc.)
gather_facts makes your playbooks smarter by giving them knowledge about the host systems.

Default: yes
Can be turned off for speed if not needed.
Example:
gather_facts: no

8. tags

Tags allow you to label tasks, plays, or roles in a playbook.
You can then run only certain parts of a playbook without executing the entire thing.
Useful when you have a large playbook with many tasks but want to execute only a subset for testing, debugging, or updates.
Example:
tags: install
Run playbook with a specific tag: --tags
ansible-playbook site.yml --tags install
Skipping Tasks by Tag : --skip-tags
ansible-playbook site.yml --skip-tags configure
Multiple Tags
tags: install,configure
ansible-playbook site.yml --tags install,configure

```
- name: Configure webserver
  hosts: webservers
  become: yes
  gather_facts: yes

  tasks:
    - name: Install Apache
      yum:
        name: httpd
        state: present
      tags: install

    - name: Start Apache service
      service:
        name: httpd
        state: started
      tags: configure

    - name: Create log directory
```

```
    file:
      path: /var/log/myapp
      state: directory
    tags: configure
```

9. loops

Used to repeat a task multiple times.
name: "{{ item }}": item is a placeholder for each value in the loop.
loop: A list of values that item will take one by one
Example:

```
tasks:
  - name: Install multiple packages
    yum:
      name: "{{ item }}"
      state: present
    loop:
      - httpd
      - mariadb
      - nginx
```

10 . roles

A structured way to organize playbooks into reusable components.
Each role contains tasks, handlers, files, templates, vars, etc.
Example:

```
roles:
  - apache
  - mysql
```

Examples for Each Ansible Playbook Keyword

1. name

Used to describe what a task or play does.

```
- name: Configure and deploy web application
  hosts: webservers
```

```
  become: yes
  tasks:
    - name: Install Git
      yum:
        name: git
        state: present

- name: Create a new directory for application logs
  file:
    path: /var/log/myapp
    state: directory
    mode: '0755'
```

---> ansible-playbook playbook1.yaml

here name is used to describe the first play "Configure and deploy web application
"
then name is used to describe about task 1 "Install Git
"
then name is used to describe about task 2 "Create a new directory for application logs "

---

2. hosts

Defines which machines the play should run on.

```
- hosts: webservers
  tasks:
    - name: Print message
      debug:
        msg: "Running on all webservers"
```

---

3. become

Used for privilege escalation (root access).

```
- hosts: all
  become: yes
  tasks:
    - name: Update package index
      yum:
        name: '*'
        state: latest
```

---

4. vars

Define variables for use in tasks.

```
- hosts: all
  vars:
    package_name: httpd

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

---

5. tasks

Contains the list of actions to execute.

```
- hosts: all
  tasks:
    - name: Create a directory
      file:
        path: /tmp/mydir
        state: directory
```

---

6. handlers

Run only when notified by a task.

```
- hosts: all
  tasks:
    - name: Update Apache config
      copy:
        src: httpd.conf
        dest: /etc/httpd/conf/httpd.conf
      notify: restart apache

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

---

7. gather_facts

Collect system information automatically.gather_facts is needed because Ansible must understand the system it is working on before running tasks.

```
- hosts: all
  gather_facts: no
  tasks:
    - name: Print a message
      debug:
        msg: "Facts gathering disabled"
```

---

8. tags

Run or skip specific tasks.

```
- hosts: all
  tasks:
    - name: Install Apache
      yum:
        name: httpd
        state: present
      tags: install

    - name: Start Apache
      service:
        name: httpd
        state: started
      tags: start
```

Run only the install task:

ansible-playbook site.yml --tags install

---

9. loops

Repeat a task multiple times.

```
- hosts: all
  tasks:
    - name: Install multiple packages
      yum:
        name: "{{ item }}"
        state: present
      loop:
        - httpd
        - mariadb
        - php
```

---

10. roles

Calls predefined roles.

```
- hosts: webservers
  roles:
    - apache
    - mysql
```

Folder structure:

```
roles/
  apache/
    tasks/main.yml
  mysql/
    tasks/main.yml
```

Main Playbook (site.yml)

```
---
- name: Install Apache and MySQL
  hosts: webservers
  become: yes

  roles:
    - apache
    - mysql
```

Role 1: Apache Role

roles/apache/tasks/main.yml

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

- name: Start Apache service
```

```
  service:
    name: httpd
    state: started
    enabled: yes
```

Role 2: MySQL Role

roles/mysql/tasks/main.yml

```
---
- name: Install MySQL (MariaDB)
  yum:
    name: mariadb-server
    state: present

- name: Start MySQL service
  service:
    name: mariadb
    state: started
    enabled: yes
```

How to Run It

ansible-playbook site.yml

Example 1:

```
---
- name: Install and Configure Apache Webserver      # name (play)
  hosts: webservers                                 # hosts
  become: yes                                       # become
  gather_facts: yes                                 # gather_facts

  vars:                                             # vars
    package_name: httpd
```

```
  tasks:                                # tasks
    - name: Install Apache package              # name (task)
      yum:
        name: "{{ package_name }}"
        state: present
      tags: install                     # tags
      notify: restart apache                    # notify handler

    - name: Start Apache service
      service:
        name: httpd
        state: started
      tags: configure

    - name: Create log directories for app
      file:
        path: "{{ item }}"
        state: directory
      loop:                             # loops
        - /var/log/myapp
        - /var/log/myapp/errors
      tags: configure

  handlers:                             # handlers
    - name: restart apache
      service:
        name: httpd
        state: restarted
```

Example 2:

Folder structure:

```
project/
  site.yml
  roles/
```

```
  deploy/
    tasks/main.yml
```

roles/deploy/tasks/main.yml

```
---
- name: Copy welcome file
  copy:
    src: welcome.txt
    dest: /home/devops/welcome.txt
```

site.yml (Main Playbook with all keywords)

```
---
- name: Create user and deploy configuration         # name (play)
  hosts: all                                  # hosts
  become: yes                                 # become
  gather_facts: no                            # gather_facts

  vars:                                       # vars
    user_name: devops

  tasks:                                      # tasks
    - name: Create a user                             # name (task)
      user:
        name: "{{ user_name }}"
        state: present
      tags: create
      notify: restart sshd                    # notify handler

    - name: Create multiple directories for user
      file:
        path: "{{ item }}"
        state: directory
        owner: "{{ user_name }}"
      loop:                                   # loops
        - "/home/{{ user_name }}/data"
        - "/home/{{ user_name }}/backup"
      tags: setup
```

```
roles:                          # roles
  - deploy

handlers:                       # handlers
  - name: restart sshd
    service:
      name: sshd
      state: restarted
```