Chapter : 03
IMPLEMENTING PLAYBOOKS

- A play is an ordered set of tasks run against hosts selected from your inventory.
- A playbook is a text file containing a list of one or more plays to run in a specific order.
- A playbook is a text file written in YAML format and is generally saved with the extension '.yml'.
- The playbook uses indentation with space characters to indicate the structure of its data.

Two basic rules for indentation & spaces are:
- Data elements at the same level in the hierarchy must have the same indentation.
- Items that are children of another item must be indented more than their parents.

~/.vimrc
→ autocmd FileType yaml setlocal ai ts=2 sw=2 et

- A playbook begins with line consisting (---) & ends with line consisting (...) ; but this is not necessary.
- An item in a YAML list starts with a single dash followed by space.
Keys in the same play must have same indentation.

The 'name' key associates an arbitrary string with the play as a label.
The 'name' key is optional but helps to document playbook.
First line of play starts with '- ' dash & space ∴ - name:

The second key is the host attribute, which specify the hostsss against which the play's tasks are run.

' hosts: hostname'
' hosts: all'
' hosts: group 1'

The last key in the play is task attribute.
The task attribute is the part of the play that actually lists, in order, the tasks to be run on the managed hosts.

The Order in which the plays and tasks are listed in a playbook is important, because ansible runs them in the same order.

Ansible-playbook command is used to run playbooks.

```
$ ansible-playbook playbook.yml
```

→ Output is generated to show the play and tasks being executed. The Output also reports the results of each task executed.

Ansible-playbook -verbosity command provides additional information.

- -v : The task results are displayed
- -vv : Task results + Task configuration
- -vvv : + information about connections to managed hosts
- -vvvv : + Connection plug-ins; users being used in the managed hosts to execute scripts.

```
$ ansible-playbook playbook.yml --syntax-check
```

→ verify syntax of a playbook.

$ ansible-playbook playbook.yml -C
→ Dry run the playbook ; changes = zero

```
- name : playbook
  hosts : all
  remote_user : abcd
  become : yes/true
  become_user : root
  become_method : sudo
```

→ overrides ansible.cfg

- Ansible-doc command is used to display detailed documentation for a module.

$ ansible-doc -l
→ To see the list of modules available on a control node.

$ ansible-doc -s yum
→ Example output that serve as a model for how to use.

The 'Metadata' section at the end of ansible-doc for the module describes its development status.

Status field:

1) stableinterface → stable.
2) preview → unstable, its keyword might change
3) depreated → will no longer available in future release
4) removed → removed ; kept for help to migrate.

The supported_by field records, who maintain the module in the upstream ansible community.

↳

1) Core
by Ansible developers upstream; always included with ansible.

2) Curated
Partners and Companies in the community; included with ansible for now. Ansible developers upstream review the curated modules for any changes.

3) Community
Not supported by core upstream developers, partners & companies.
Maintained by general open source community.

→ Playbook Variations ←——

→ '#' is used to comment a line in a playbook
→ No need of quotation marks '' "" for strings.
→ To write multiline strings;

```
include_newline: |
        line 1
        line 2
        line 3
```

(pipe '|' symbol can be used also '>' symbol does the same)

→
```
name: svcrole
svcservice: httpd
svcport: 80
```
⇒ {name: ___, svcservice: ___, svcport: ___}

→
```
hosts:
  - node1
  - node2
```
⇒ hosts: [node1, node2]

→
```
service:
  name: httpd
  enabled: true
  state: started
```
⇒ service: name=httpd enable=true.

M T W T F S S
Page No.: 14
Date:
YOUVA

Modules to study:
1) yum
2) copy
3) service
4) firewalld
5) uri
6) get_url