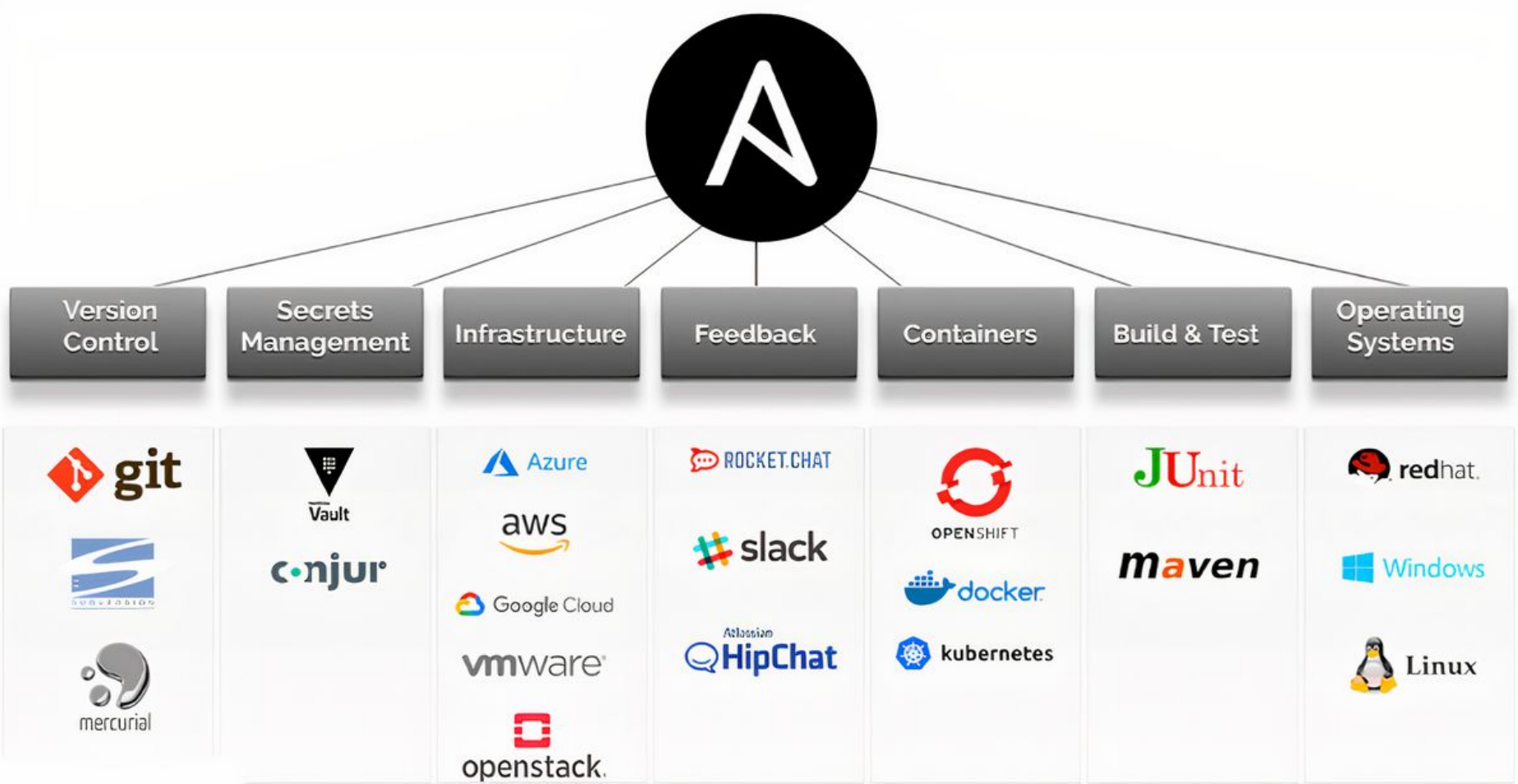
SDJ INTERNATIONAL COLLEGE

# Web Development Operations
## Unit 3

*– By Tushar Sir*




*– By Tushar Sir*

# Ansible: Introduction and working

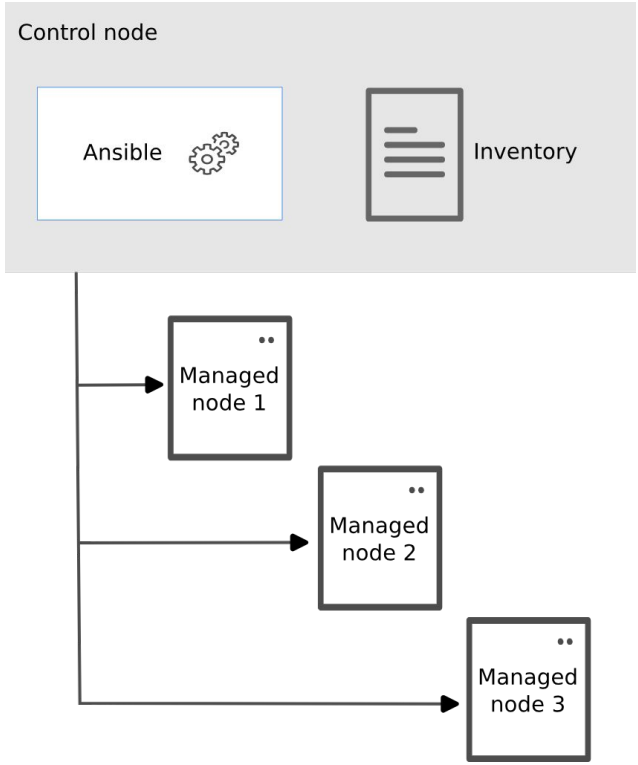
- **Ansible** is an open-source, command-line IT automation software application. It's written in Python and can be used for: Configuring systems, Deploying software, Orchestrating advanced workflows, Task automation, IT orchestration.
- Ansible is easy to set up and runs on Linux, Mac, or BSD. It's commonly used in DevOps workstreams for configuration management.
- Ansible works by:
  - Connecting to nodes and pushing out small programs called "Ansible modules".
  - Executing these modules (over SSH by default).
  - Removing the modules when finished.
  - Using playbooks written in the simple language YAML to translate what you define as true in your infrastructure into actions your systems must take to meet that definition.

***– By Tushar Sir***

# Ansible: Introduction and working

Ansible automates the management of remote systems and controls their desired state. A basic Ansible environment has three main components:

- **Control node :** A system on which Ansible is installed. You run Ansible commands such as ansible or ansible-inventory on a control node.
- **Managed node :** A remote system, or host, that Ansible controls.
- **Inventory :** A list of managed nodes that are logically organized. You create an inventory on the control node to describe host deployments to Ansible.




***– By Tushar Sir***

# Ansible: Introduction and working

- Ansible uses playbook to describe automation jobs, and playbook uses very simple language i.e. **YAML - "Yet another markup language"** (It's a human-readable data serialization language & is commonly used for configuration files, but could be used in many applications where data is being stored) which is very easy for humans to understand, read and write.
- Hence the advantage is that even the IT infrastructure support guys can read and understand the playbook and debug if needed.
- Ansible is designed for multi-tier deployment.
- Ansible does not manage one system at time, it models IT infrastructure by describing all of your systems are interrelated.
- Ansible is completely agentless which means Ansible works by connecting your nodes through SSH(by default).

***– By Tushar Sir***

# Benefits of Ansible

Ansible provides DevOps engineers with four clear benefits :

- **It reduces the resources needed for managing IT -** Sysadmins can contain hundreds or even thousands of machines from a single point and in a single go.
- **It makes automation accessible -** One of the design goals of Ansible was that minimal learning should be required to use it. The platform uses YAML(Yet Another Markup Language), a human-readable language with elements from other common programming languages.
- **The Ansible platform doesn't affect performance -** Ansible doesn’t require agents or software running or installed in managed systems. Because of this, managed systems don’t need to spend computational resources on Ansible.
- **It ensures consistency -** The platform was designed to be minimal and enable users to create consistent environments. And the entire operation is carried over an SSH connection means the forum doesn’t add more complexity to the systems.

***– By Tushar Sir***

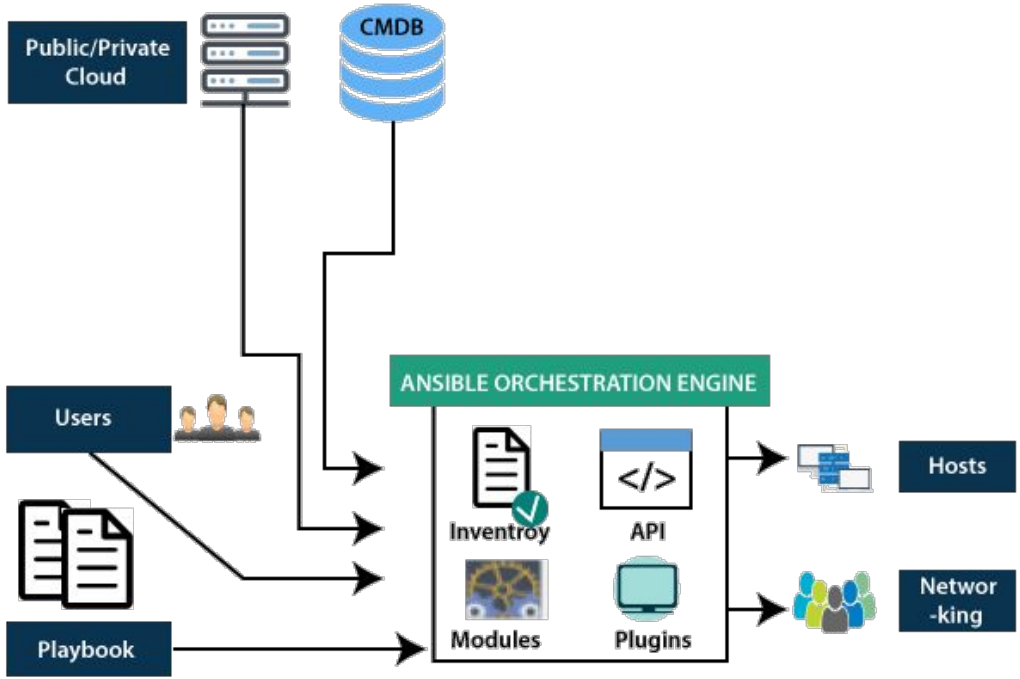
# How does Ansible work?

- Ansible works by executing ‘Modules’ or code in target machines referred to as managed nodes.
- Modules, once executed, are removed from the target systems. These Modules can be for anything from copying a file to a remote machine and creating a compressed archive of files to managing VLAN interfaces.
- In Ansible, a Task is a simple action applied to target machines. Tasks use Modules to execute an action on a remote device.
- These Tasks are further rolled into an ordered list in a Play. A Play defines which all tasks are applied to which all machines.

***– By Tushar Sir***

# Ansible Architecture

- The Ansible orchestration engine interacts with a user who is writing the Ansible playbook to execute the Ansible orchestration and interact along with the services of private or public cloud and configuration management database.




*– By Tushar Sir*

# Ansible Architecture

**Inventory :**

- Inventory is lists of nodes or hosts having their IP addresses, databases, servers, etc. which are need to be managed.

**API's :**

- The Ansible API's works as the transport for the public or private cloud services.

**Modules :**

- Ansible connected the nodes and spread out the Ansible modules programs. Ansible executes the modules and removed after finished. These modules can reside on any machine; no database or servers are required here. You can work with the chose text editor or a terminal or version control system to keep track of the changes in the content.

***– By Tushar Sir***

# Ansible Architecture

**Plugins :**

- Plugins is a piece of code that expends the core functionality of Ansible. There are many useful plugins, and you also can write your own.

**Playbooks :**

- Playbooks consist of your written code, and they are written in YAML format, which describes the tasks and executes through the Ansible. Also, you can launch the tasks synchronously and asynchronously with playbooks.

**Hosts :**

- In the Ansible architecture, hosts are the node systems, which are automated by Ansible, and any machine such as RedHat, Linux, Windows, etc.

***– By Tushar Sir***

# Ansible Architecture

**Networking :**

- Ansible is used to automate different networks, and it uses the simple, secure, and powerful agentless automation framework for IT operations and development. It uses a type of data model which separated from the Ansible automation engine that spans the different hardware quite easily.

**Cloud :**

- A cloud is a network of remote servers on which you can store, manage, and process the data. These servers are hosted on the internet and storing the data remotely rather than the local server. It just launches the resources and instances on the cloud, connect them to the servers, and you have good knowledge of operating your tasks remotely.

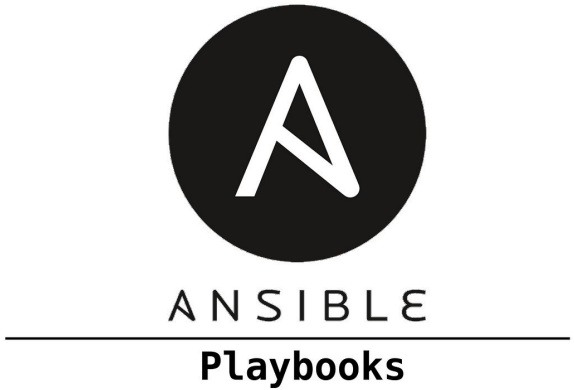
**CMDB :**

- CMDB is a type of repository which acts as a data warehouse for the IT installations.

***– By Tushar Sir***

# Ansible playbooks

- All the **Plays** are combined to form what's called a **Playbook**. This is where users write all Ansible code.
- The Ansible playbook is executed on a control node, and the Plays are executed in the order in which it is written. The Tasks within the Plays are, in turn, performed similarly.
- When executing a Task, the control node sends the respective Modules to all the target machines (managed nodes) mapped to the task.
- Ansible Modules or, in turn, Ansible Playbooks are generally idempotent. This means that before the Module is executed, it checks if the managed node is already at the desired state. If so, no action is performed there.




***– By Tushar Sir***

# YAML

- **YAML** ***("Yet another markup language" / "YAML ain't markup language")*** is a human-readable data serialization language. It's often used to write configuration files. YAML is a strict superset of JSON, which means it can do everything that JSON can and more.
- YAML is designed to be directly writable and readable by humans. It has the following features:
  - It's a strict superset of JSON, with the addition of syntactically significant newlines and indentation.
  - It natively encodes scalars (such as strings, integers, and floats), lists, and associative arrays (also known as maps, dictionaries or hashes).
  - It allows custom data types.
  - It can be authored in any text editor.

***– By Tushar Sir***

# YAML


**Rules for Creating YAML file**

- When you are creating a file in YAML, you should remember the following basic rules −
- YAML is case sensitive.
- The files should have *.yaml* as the extension.
- YAML does not allow the use of tabs while creating YAML files; spaces are allowed instead

```yaml
title: YAML Ain't Markup Language
name: Mohsin Zaheer
profession: Software Engineer
learn:
  - Basic Data Structure
  - Commeting
```

***– By Tushar Sir***

# Basic Components of YAML File

The basic components of YAML are described below −

**Conventional Block Format :**

- This block format uses ***hyphen+space*** to begin a new item in a specified list. Observe the example shown below −

```
--- # Favorite movies
 - Casablanca
 - North by Northwest
 - The Man Who Wasn't There
```

**Inline Format :**

- Inline format is delimited with comma and space and the items are enclosed in JSON. Observe the example shown below −

```
--- # Shopping list
  [milk, groceries, eggs, juice, fruits]
```

***– By Tushar Sir***

# Basic Components of YAML File

**Folded Text :**

- Folded text converts newlines to spaces and removes the leading whitespace. Observe the example shown below −

```
- {name: John Smith, age: 33}
- name: Mary Smith
  age: 27
```

- The structure which follows all the basic conventions of YAML is shown below −

```
men: [John Smith, Bill Jones]
women:
  - Mary Smith
  - Susan Williams
```

***– By Tushar Sir***

# Synopsis of YAML Basic Elements

- The synopsis of YAML basic elements is given here: Comments in YAML begins with the (#) character.
- Comments must be separated from other tokens by whitespaces.
- Indentation of whitespace is used to denote structure.
- Tabs are not included as indentation for YAML files.
- List members are denoted by a leading hyphen (-).
- List members are enclosed in square brackets and separated by commas.
- Associative arrays are represented using colon ( : ) in the format of key value pair. They are enclosed in curly braces {}.
- Multiple documents with single streams are separated with 3 hyphens (---).
- Repeated nodes in each file are initially denoted by an ampersand (&) and by an asterisk (*) mark later.
- YAML always requires colons and commas used as list separators followed by space with scalar values.
- Nodes should be labelled with an exclamation mark (!) or double exclamation mark (!!), followed by string which can be expanded into an URI or URL.

***– By Tushar Sir***

# Key, Value in YAML

- In YAML, key-value pairs are the basic building blocks for storing data.
- The key is always a string, and the value can be any data type. The key and value are separated by a colon.
- The value can be:
  - A **string**, A **number**, A **float**, A **boolean**, A **list**, Another key-value pair.
- Here are some conventions for using keys in YAML:
  - Use brackets when the key is a list or an array
  - Use curly braces when the key is a dictionary or an object
  - Use the pipe character when the key is a multi-line string
  - YAML uses indentation to create nested structures, such as lists and dictionaries.

***– By Tushar Sir***

# Key, Value in YAML

YAML

```yaml
apis:
  - name: login
    port: 8080
  - name: profile
    port: 8090
```

XML

```xml
<apis>
    <api>
        <name>login</name>
        <port>8080</port>
    </api>
    <api>
        <name>profile</name>
        <port>8090</port>
    </api>
</apis>
```

JSON

```json
{
  "apis": [
    {
      "name": "login",
      "port": 8080
    },
    {
      "name": "profile",
      "port": 8090
    }
  ]
}
```

*– By Tushar Sir*

# List, List inside Dictionaries, List of Dictionaries

- **Dictionaries** and **lists** can also be represented in an abbreviated form if you really want to:

```
martin: {name: Martin D'vloper, job: Developer, skill: Elite}
fruits: ['Apple', 'Orange', 'Strawberry', 'Mango']
```

- These are called “Flow collections”.
- Ansible doesn’t really use these too much, but you can also specify a **boolean** value (true/false) in several forms:

```
create_key: true
needs_agent: false
knows_oop: True
likes_emacs: TRUE
uses_cvs: false
```

***– By Tushar Sir***

# Ansible Installation process

- **Installing Ansible on Ubuntu**

```
$ sudo apt update
$ sudo apt install software-properties-common
$ sudo add-apt-repository --yes --update ppa:ansible/ansible
$ sudo apt install ansible
```

- **Installing Ansible on Windows**

Install Ubuntu on your Windows Machine : -
https://ubuntu.com/tutorials/install-ubuntu-on-wsl2-on-windows-10

Then repeat steps of installing Ansible on Ubuntu.

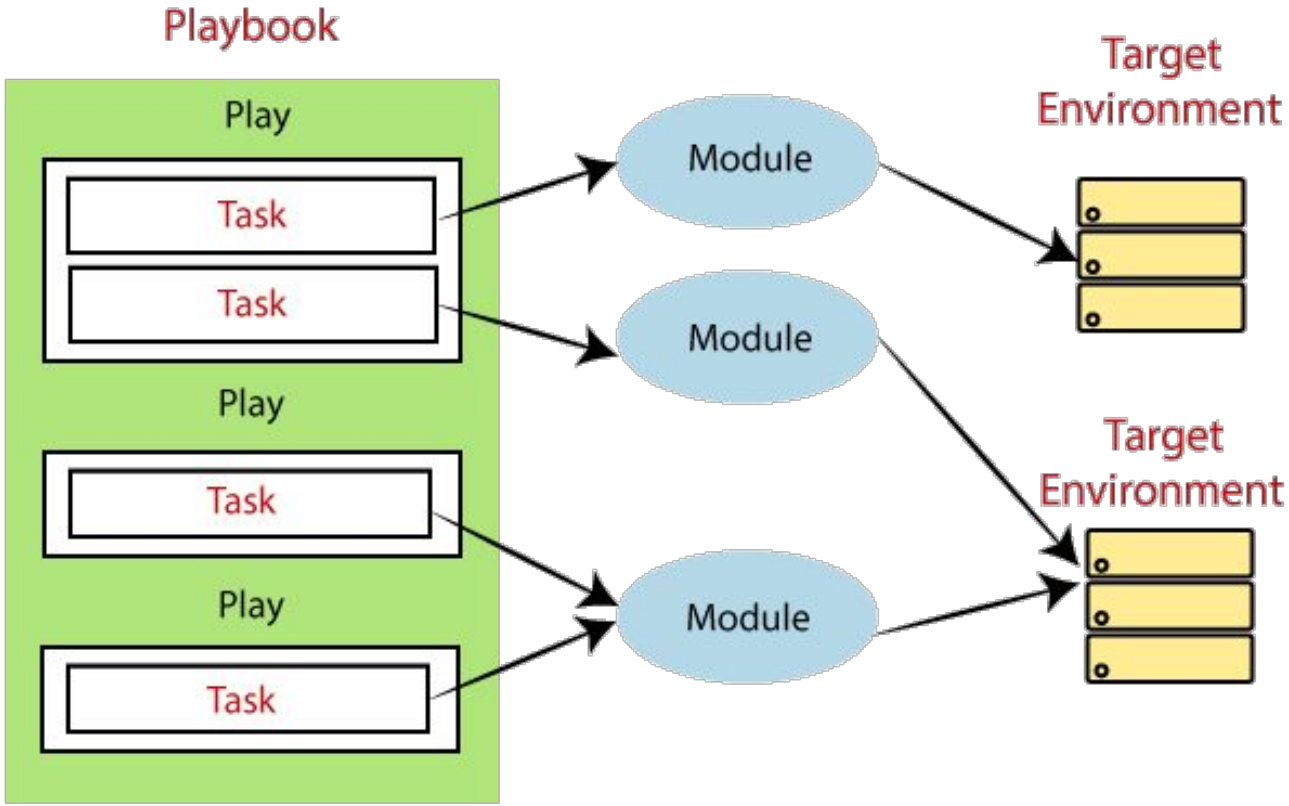
***– By Tushar Sir***

# Concept of Ansible Playbook

- An Ansible playbook is a file that contains instructions for automating tasks on remote hosts. Playbooks are written in YAML, a human-readable markup language.
- A playbook typically consists of one or more plays, which are collections of tasks run in sequence. A play is an ordered grouping of tasks mapped to specific hosts. Tasks in a playbook are executed from top to bottom.
- Playbooks are the simplest way in Ansible to automate repeating tasks. They are reusable and consistent configuration files.
- Ansible is a configuration management tool from Red Hat that automates the process of configuring multiple servers and deploying applications.

***– By Tushar Sir***

# Concept of Ansible Playbook




*– By Tushar Sir*

# Concept of Ansible Playbook

- Each playbook is a collection of one or more plays. Playbooks are structured by using Plays. There can be more than one play inside a playbook.
- The function of the play is to map a set of instructions which is defined against a particular host.
- Playbooks can:
  - declare configurations.
  - orchestrate steps of any manual ordered process, on multiple sets of machines, in a defined order.
  - launch tasks synchronously or asynchronously.

***– By Tushar Sir***

# Create Playbook

- Let's start by writing an example YAML file. First, we must define a task. These are the interface to ansible modules for roles and playbooks.
- One playbook with one play, containing multiple tasks looks like the below example:

```
name: install and configure DB
hosts: testServer
become: yes

vars:
  oracle_db_port_value : 1521

tasks:
-name: Install the Oracle DB
  yum: <code to install the DB>

-name: Ensure the installed service is enabled and running
service:
  name: <your service name>
```

- Above is a basic syntax of a playbook. Save it in a file as test.yml. A YAML syntax needs to follow the correct indentation.

***– By Tushar Sir***

# YAML Tags

| Tags | Explanation |
|---|---|
| **Name** | It specifies the name of the Ansible Playbooks. |
| **Hosts** | It specifies the lists of the hosts against which you want to run the task. And the host's Tag is mandatory. It tells Ansible that on which hosts to run the listed tasks. These tasks can be run on the same machine or the remote machine. One can run the tasks on the multiple machines, and the host's tag can have a group of host's entry as well. |
| **Vars** | Vars tag defines the variables which you can use in your playbook. Its usage is similar to the variables in any programming language. |
| **Tasks** | Tasks are the lists of the actions which need to perform in the playbooks. All the playbooks should contain the tasks to be executed. A task field includes the name of the task. It is not mandatory but useful for debugging the playbook. Internally each task links to a piece of code called a module. A module should be executed, and arguments that are required for the module you want to run. |

***– By Tushar Sir***

# Thank You

- **By Tushar Sir**