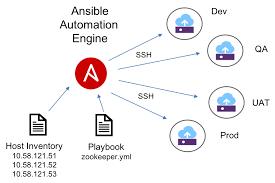
Ansible is a radically simple IT automation engine that automates [cloud provisioning](https://www.ansible.com/provisioning?hsLang=en-us), [configuration management](https://www.ansible.com/configuration-management?hsLang=en-us), [application deployment](https://www.ansible.com/application-deployment?hsLang=en-us), [intra-service orchestration](https://www.ansible.com/orchestration?hsLang=en-us), and many other IT needs.

Designed for multi-tier deployments since day one, Ansible models your IT infrastructure by describing how all of your systems inter-relate, rather than just managing one system at a time.

It uses no agents and no additional custom security infrastructure, so it's easy to deploy - and most importantly, it uses a very simple language (YAML, in the form of Ansible Playbooks) that allow you to describe your automation jobs in a way that approaches plain English.



What ?

Ansible is an open source IT Management tool use to deployment and orchestration tool.It aims to provide large productivity gains to a wide variety of automation.

History:

Michel Dehaan developed Ansible and the ansible project began in feb 2012.

RedHat acquired the ansible tool in 2015.

Ansible is available for RHEL Debain centos oracle . Linux, windows powershell.

Can we use this tool whether your servers are in the cloud.

It support yaml(yet another markup language)

Ansible is agentless. direct commumnicate with nodes.so it is secure.

Advantageous:

Ansible is open source used by everyone.

It is very secure due to agentless capabilities an open ssh security features

Ansible does not need any special system administrator skills to install and use it.

It support push mechanisum

Disadvantagous:

Cannot achive full automation.

Insufficient user interface,through ansible tower(Enterprise version) GUI.but it in still development stage

New to the market therefore limited support and documentis available.

Main terms In Ansible:

**Ansible Server:**

The machine where ansible is installed and form which all task and playbook will be run.

Module:

Basically ,a module is a command or set of similar command meet to be executed on the client side

Task:

A task is a section that consist of a single procedure to be completed.

Role:

A way of organizing tasks and related files to be later called in playbook.

## Creating a New Role

The directory structure for roles is essential to create a new role.

### **Role Structure**

Roles have a structured layout on the file system. The default structure can be changed but for now let us stick to defaults.

Each role is a directory tree in itself. The role name is the directory name within the /roles directory.

$ ansible-galaxy -h

### **Usage**

ansible-galaxy [delete|import|info|init|install|list|login|remove|search|setup] [--help] [options] ...

### **Options**

* **-h, --help** − Show this help message and exit.
* **-v, --verbose** − Verbose mode (-vvv for more, -vvvv to enable connection debugging)
* **--version** − Show program's version number and exit.

### **Creating a Role Directory**

The above command has created the role directories.

$ ansible-galaxy init vivekrole

ERROR! The API server (https://galaxy.ansible.com/api/) is not responding, please try again later.

$ ansible-galaxy init --force --offline vivekrole

- vivekrole was created successfully

$ tree vivekrole/

Fact:

Information fetched from the client system from the global variable with the gather fact opretion.

Play:

Execution of playbook

Handler:

Task which is cell only if a notifier is present

Notifier:

Section attributes to a task which called a handler if the output manage

Playbook:

It consist code in yaml format which describe task be executed

Host:Nodes ,which are automated by ansible

Variable in playbooks are **very similar** to using variables in any programming language. It helps you to use and assign a value to a variable and use that anywhere in the playbook. One can put conditions around the value of the variables and accordingly use them in the playbook.

### **Example**

- hosts : <your hosts>

vars:

tomcat\_port : 8080

In the above example, we have defined a variable name **tomcat\_port** and assigned the value 8080 to that variable and can use that in your playbook wherever needed.

Now taking a reference from the example shared. The following code is from one of the roles (install-tomcat) −

block:

- name: Install Tomcat artifacts

action: >

yum name = "demo-tomcat-1" state = present

register: Output

always:

- debug:

msg:

- "Install Tomcat artifacts task ended with message: {{Output}}"

- "Installed Tomcat artifacts - {{Output.changed}}"

Here, the output is the variable used.

Let us walk through all the keywords used in the above code −

* **block** − Ansible syntax to execute a given block.
* **name** − Relevant name of the block - this is used in logging and helps in debugging that which all blocks were successfully executed.
* **action** − The code next to action tag is the task to be executed. The action again is a Ansible keyword used in yaml.
* **register** − The output of the action is registered using the register keyword and Output is the variable name which holds the action output.
* **always** − Again a Ansible keyword , it states that below will always be executed.
* **msg** − Displays the message.

### **Usage of variable - {{Output}}**

This will read the value of variable Output. Also as it is used in the msg tab, it will print the value of the output variable.

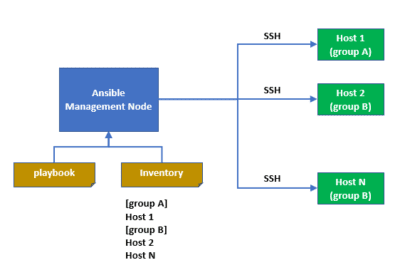
Additionally, you can use the sub properties of the variable as well. Like in the case checking {{Output.changed}} whether the output got changed and accordingly use it.

## Exception Handling in Playbooks

Exception handling in Ansible is similar to exception handling in any programming language. An example of the exception handling in playbook is shown below.

tasks:

**ow Ansible Works?**



Ansible works by connecting to nodes

Ansible works by connecting to nodes and pushing out small programs called as ansible modules. Ansible then executes these modules over SSH by default and then remove them when finished.

Ansible management node is the controlling node, which controls the entire execution of the Playbook. It’s the node from which you are running the installation, and the inventory file provides the list of the host where the modules need to be run. The management node makes ssh connection, and then it executes the modules on the host machines and installs the product. It removes the modules once they are installed. So that’s how ansible works.

Next, you may be interested in learning how to Install and configure Ansible.