Ad hoc commands are commands which can be run individually to perform quick functions. These commands need not be performed later.

For example, you have to reboot all your company servers. For this, you will run the Adhoc commands from '**/usr/bin/ansible**'.

These ad-hoc commands are not used for configuration management and deployment, because these commands are of one time usage.

ansible-playbook is used for configuration management and deployment.

Parallelism and Shell Commands

Reboot your company server in 12 parallel forks at time. For this, we need to set up SSHagent for connection.

```
$ ssh-agent bash
$ ssh-add ~/.ssh/id_rsa
```

To run reboot for all your company servers in a group, 'abc', in 12 parallel forks −

```
$ Ansible abc -a "/sbin/reboot" -f 12
```

By default, Ansible will run the above Ad-hoc commands form current user account. If you want to change this behavior, you will have to pass the username in Ad-hoc commands as follows −

```
$ Ansible abc -a "/sbin/reboot" -f 12 -u username
```

File Transfer

You can use the Ad-hoc commands for doing **SCP** (Secure Copy Protocol) lots of files in parallel on multiple machines.

Transferring file to many servers/machines

```
$ Ansible abc -m copy -a "src = /etc/yum.conf dest = /tmp/yum.conf"
```

Creating new directory

```
$ Ansible abc -m file -a "dest = /path/user1/new mode = 777 owner =
user1 group = user1 state = directory"
```

Deleting whole directory and files

```
$ Ansible abc -m file -a "dest = /path/user1/new state = absent"
```

Managing Packages

The Ad-hoc commands are available for yum and apt. Following are some Ad-hoc commands using yum.

The following command checks if yum package is installed or not, but does not update it.

```
$ Ansible abc -m yum -a "name = demo-tomcat-1 state = present"
```

The following command check the package is not installed.

```
$ Ansible abc -m yum -a "name = demo-tomcat-1 state = absent"
```

The following command checks the latest version of package is installed.

```
$ Ansible abc -m yum -a "name = demo-tomcat-1 state = latest"
```

Gathering Facts

Facts can be used for implementing conditional statements in playbook. You can find adhoc information of all your facts through the following Ad-hoc command −

```
$ Ansible all -m setup
```

Playbooks in Ansible.

Playbooks are the files where Ansible code is written. Playbooks are written in YAML format. YAML stands for Yet Another Markup Language. **Playbooks** are one of the core features of Ansible and tell Ansible what to execute. They are like a to-do list for Ansible that contains a list of tasks.

Playbooks contain the steps which the user wants to execute on a particular machine. Playbooks are run sequentially. Playbooks are the building blocks for all the use cases of Ansible.

Playbook Structure

Each playbook is an aggregation of one or more plays in it. Playbooks are structured using Plays. There can be more than one play inside a playbook.

The function of a play is to map a set of instructions defined against a particular host.

YAML is a strict typed language; so, extra care needs to be taken while writing the YAML files. There are different YAML editors but we will prefer to use a simple editor like notepad++. Just open notepad++ and copy and paste the below yaml and change the language to YAML (Language → YAML).

A YAML starts with --- (3 hyphens)

Create a Playbook

Let us start by writing a sample YAML file. We will walk through each section written in a yaml file.

```
---
   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>
```

The above is a sample Playbook where we are trying to cover the basic syntax of a playbook. Save the above content in a file as **test.yml**. A YAML syntax needs to follow the correct indentation and one needs to be a little careful while writing the syntax.

The Different YAML Tags

Let us now go through the different YAML tags. The different tags are described below –

name

This tag specifies the name of the Ansible playbook. As in what this playbook will be doing. Any logical name can be given to the playbook.

hosts

This tag specifies the lists of hosts or host group against which we want to run the task. The hosts field/tag is mandatory. It tells Ansible on which hosts to run the listed tasks. The tasks can be run on the same machine or on a remote machine. One can run the tasks on multiple machines and hence hosts tag can have a group of hosts' entry as well.

vars

Vars tag lets you define the variables which you can use in your playbook. Usage is similar to variables in any programming language.

tasks

All playbooks should contain tasks or a list of tasks to be executed. Tasks are a list of actions one needs to perform. A tasks field contains the name of the task. This works as the help text for the user. It is not mandatory but proves useful in debugging the playbook.

Each task internally links to a piece of code called a module. A module that should be executed, and arguments that are required for the module you want to execute.

Roles provide a framework for fully independent, or interdependent collections of variables, tasks, files, templates, and modules.

In Ansible, the role is the primary mechanism for breaking a playbook into multiple files. This simplifies writing **complex playbooks**, and it makes them easier to reuse. The breaking of playbook allows you to logically break the playbook into reusable components.

Each role is basically limited to a particular functionality or desired output, with all the necessary steps to provide that result either within that role itself or in other roles listed as dependencies.

Roles are not playbooks. Roles are small functionality which can be independently used but have to be used within playbooks. There is no way to directly execute a role. Roles have no explicit setting for which host the role will apply to.

Top-level playbooks are the bridge holding the hosts from your inventory file to roles that should be applied to those hosts.

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/
vivekrole/
├── defaults
│   └── main.yml
├── files ├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md ├── tasks
│   └── main.yml
├── templates ├── tests │    ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files
```

Utilizing Roles in Playbook

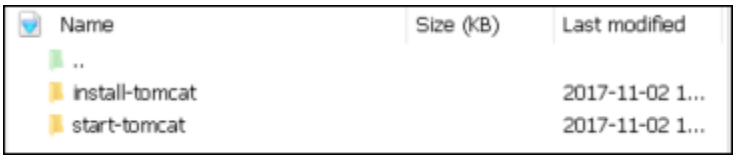
This is the code of the playbook we have written for demo purpose. This code is of the playbook vivek_orchestrate.yml. We have defined the hosts: **tomcat-node** and called the two roles – **install-tomcat** and **start-tomcat**.

The problem statement is that we have a war which we need to deploy on a machine via Ansible.

```
---
- hosts: tomcat-node
roles:
   - {role: install-tomcat}
   - {role: start-tomcat}
```

Contents of our directory structure from where we are running the playbook.


```
$ ls
ansible.cfg  hosts  roles  vivek_orchestrate.retry
vivek_orchestrate.yml
```



There is a tasks directory under each directory and it contains a main.yml. The main.yml contents of install-tomcat are −

```
---
#Install vivek artifacts
-
   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}}"
```

The contents of main.yml of the start tomcat are −

```
#Start Tomcat
-
   block:
      - name: Start Tomcat
      command: <path of tomcat>/bin/startup.sh"
      register: output
      become: true

   always:
      - debug:
```

```
        msg:
           - "Start Tomcat task ended with message: {{output}}"
           - "Tomcat started - {{output.changed}}"
```

The advantage of breaking the playbook into roles is that anyone who wants to use the Install tomcat feature can call the Install Tomcat role.

Breaking a Playbook into a Role

If not for the roles, the content of the main.yml of the respective role can be copied in the playbook **yml** file. But to have modularity, roles were created.

Any logical entity which can be reused as a reusable function, that entity can be moved to role. The example for this is shown above

Ran the command to run the playbook.

```
-vvv option for verbose output – verbose output
$ cd vivek-playbook/
```

This is the command to run the playbook

```
$ sudo ansible-playbook -i hosts vivek_orchestrate.yml –vvv
---------------------------------------------------------------------
---------------------------------------------------------------------
----
```

Output

The generated output is as seen on the screen −

Using **/users/demo/vivek-playbook/ansible.cfg** as config file.

```
PLAYBOOK: vivek_orchestrate.yml
*****************************************************
*************************************************************
1 plays in vivek_orchestrate.yml

PLAY [tomcat-node]
*********************************************************************
***
******** ***********************************************

TASK [Gathering Facts]
***********************************************
*****************************
*****************************************
Tuesday 21 November 2017  13:02:05 +0530 (0:00:00.056) 0:00:00.056
******
Using module file
/usr/lib/python2.7/sitepackages/ansible/modules/system/setup.py
<localhost> ESTABLISH LOCAL CONNECTION FOR USER: root
<localhost> EXEC /bin/sh -c 'echo ~ && sleep 0'
```

```
<localhost> EXEC /bin/sh -c '( umask 77 && mkdir -p "` echo
   /root/.ansible/tmp/ansible-tmp-1511249525.88-259535494116870 `"
&&
   echo ansible-tmp-1511249525.88-259535494116870="`
   echo /root/.ansible/tmp/ansibletmp-1511249525.88-259535494116870
`" ) && sleep 0'
<localhost> PUT /tmp/tmpPEPrkd TO
   /root/.ansible/tmp/ansible-tmp-
1511249525.88259535494116870/setup.py
<localhost> EXEC /bin/sh -c 'chmod u+x
   /root/.ansible/tmp/ansible-tmp1511249525.88-259535494116870/
   /root/.ansible/tmp/ansible-tmp-
1511249525.88259535494116870/setup.py && sleep 0'
<localhost> EXEC /bin/sh -c '/usr/bin/python
   /root/.ansible/tmp/ansible-tmp1511249525.88-
259535494116870/setup.py; rm -rf
   "/root/.ansible/tmp/ansible-tmp1511249525.88-259535494116870/" >
/dev/null 2>&1 && sleep 0'
ok: [server1]
META: ran handlers
TASK [install-tomcat : Install Tomcat artifacts]
**********************************
*********************************************************************
task path: /users/demo/vivek-playbook/roles/install-
tomcat/tasks/main.yml:5
Tuesday 21 November 2017  13:02:07 +0530 (0:00:01.515)
0:00:01.572 ******
Using module file
/usr/lib/python2.7/sitepackages/ansible/modules/packaging/os/yum.py
<localhost> ESTABLISH LOCAL CONNECTION FOR USER: root
<localhost> EXEC /bin/sh -c 'echo ~ && sleep 0'
<localhost> EXEC /bin/sh -c '( umask 77 && mkdir -p "` echo
   /root/.ansible/tmp/ansible-tmp-1511249527.34-40247177825302 `"
&& echo
   ansibletmp-1511249527.34-40247177825302="` echo
   /root/.ansible/tmp/ansible-tmp1511249527.34-40247177825302 `" )
&& sleep 0'
<localhost> PUT /tmp/tmpu83chg TO
   /root/.ansible/tmp/ansible-tmp-
1511249527.3440247177825302/yum.py
<localhost> EXEC /bin/sh -c 'chmod u+x
   /root/.ansible/tmp/ansible-tmp1511249527.34-40247177825302/
   /root/.ansible/tmp/ansible-tmp-
1511249527.3440247177825302/yum.py && sleep 0'
<localhost> EXEC /bin/sh -c '/usr/bin/python
   /root/.ansible/tmp/ansible-tmp1511249527.34-
40247177825302/yum.py; rm -rf
   "/root/.ansible/tmp/ansible-tmp1511249527.34-40247177825302/" >
/dev/null 2>
```

```
   &1 && sleep 0'
changed: [server1] => {
   "changed": true,
   "invocation": {
      "module_args": {
         "conf_file": null,
         "disable_gpg_check": false,
         "disablerepo": null,
         "enablerepo": null,
"exclude": null,
         "install_repoquery": true,
         "installroot": "/",
         "list": null,
         "name": ["demo-tomcat-1"],
         "skip_broken": false,
         "state": "present",
         "update_cache": false,
         "validate_certs": true
      }
   },
   "msg": "",
   "rc": 0,
   "results": [
      "Loaded plugins: product-id,
      search-disabled-repos,
      subscriptionmanager\nThis system is not registered to Red Hat
Subscription Management.
      You can use subscription-manager to register.\nResolving
Dependencies\n-->
      Running transaction check\n--->
      Package demo-tomcat-1.noarch 0:SNAPSHOT-1 will be
installed\n--> Finished Dependency
      Resolution\n\nDependencies Resolved\n

\n================================================================
==============\n
      Package Arch Version Repository

Size\n================================================================
=====\nInstalling:\n
      demo-tomcat-1 noarch SNAPSHOT-1 demo-repo1 7.1
M\n\nTransaction

Summary\n================================================================
========\nInstall  1
      Package\n\nTotal download size: 7.1 M\nInstalled size: 7.9
M\nDownloading
         packages:\nRunning transaction
```

Hit the following URL and you will be directed to a page as shown below
– **http://10.76.0.134:11677/HelloWorld/HelloWorld**