AKSHIT MITTAL
10/20/23

ANSIBLE Playbooks, Modules, Registers, Variables, Fact Variables, Prompt and other Functions, Loops, Tags, Handlers

Links:- [All modules — Ansible Documentation]
Terms:-

- YAML (Yaml Ain't a Markup Language)- Ansible playbooks are written in YAML. YAML is nothing but a set of Key: Values, use to define various rules and configurations, and perform various tasks. A Key can contain one or more objects and they can have multiple values, defined in the form of lists.

```
user: root
vars:
  - var1
  - var2
  - var3
tasks:
  - name: this is my first object
    task1: install java
  - name: this is second object
    tasks2: install maven
  - name: this is 3rg object
    tasks3: instll jenkins
```

```
Linux commands:
 adduser demouser
 password demouser
 group demouser
 owner root
 uid  2345
 home: /home/demouser


Write a YAML code that will create 2 users with aboive provided data

Your key is Users:
```

```
Your key is Users:

---
user:
- user1:
   adduser: demouser
   password: demouser
   group: demouser
   owner: root
   uid: 123
   home: /home/demouser
- user2:
 adduser: demouser
 password: demouser
 group: demouser
 owner: root
 uid: 123
 home: /home/demouser
```

- Playbooks- It is the file defining a queue or a set of tasks that to is be executed. A playbook can have multiple plays doing different functions on different hosts. It consist of:
  name – Name of the play, you want to set
  hosts – Name of the host on which the task is to be executed. Ansible Check Inventory file for the hostnames.
  tasks – Tasks section containing modules and parameters with values
  Name – Name of Task you want to set
  modulename – name of the module
  par1 , value – parameter of module with value given.

  Please note* You should use ":" sign to define key value pair in the playbook in task section but can use "=" sign to define key values pair in the adhoc commands.

```
- name: play1
  hosts: webserver
  tasks:
  - name: Execute tasks1
    moduleName: par1=value par2=value
  - name: Execute task2 on webserver
    moduleName: par1=value par2=value


- name: play2
  hosts: dbserver
  tasks:
  - name: Execute tasks1
    moduleName: par1=value par2=value
  - name: Execute task2 on webserver
    moduleName: par1=value par2=value


- name: play3
  hosts: javaapp
  tasks:
  - name: Execute tasks1
    moduleName: par1=value par2=value
  - name: Execute task2 on webserver
    moduleName: par1=value par2=value
```

1. Create the First playbook in Ansible
   → The playbook will be written in the Ansible Controller Machine. As it is installed on the AWS EC2 machine, we will operate machine on our local system. We will use ssh-client url to run EC2 controller machine on our system.
   → Switch to ansiuser by providing the password for the ansiuser → Change directory to /home/ansiuser where all our inventory and config files are present.
   → Create the first ansible playbook by the extension .yml → It will be according to what was defined in the earlier sections
   → Ansible Modules Link - [debug – Print statements during execution — Ansible Documentation]
   → Run #ansible-playbook Filename –syntax—check - to check the syntax of your playbook. → Run the playbook with command #ansible-playbook FileName.

```
admin@DESKTOP-9UJRCUE MINGW64 ~/Downloads
$ ssh -i "AnsibleKeyPair.pem" ubuntu@ec2-65-2-191-20.ap-south-1.compute.amazonaws.
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1036-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri Oct 20 12:38:42 UTC 2023

  System load:  0.0               Processes:             123
  Usage of /:   36.3% of 7.57GB   Users logged in:       1
  Memory usage: 33%               IPv4 address for eth0: 172.31.33.110
```

```
ubuntu@AnsibleControllerMachine:~$ su ansiuser
Password:
```

```
ansiuser@AnsibleControllerMachine:/home/ubuntu$ cd ..
ansiuser@AnsibleControllerMachine:/home$ ls
ansiuser  ubuntu
ansiuser@AnsibleControllerMachine:/home$ su ansiuser/
su: user ansiuser/ does not exist
ansiuser@AnsibleControllerMachine:/home$ cd ansiuser/
ansiuser@AnsibleControllerMachine:~$ ls
```

ansiuser@AnsibleControllerMachine: ~

```
- name: FirstPlay
  hosts: webserver
  tasks:
  - name: Task1_Debug_Module
    debug: msg="My Worker Node is executed"
~
~
~
~
~
~
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml -sytanx--check
usage: ansible-playbook [-h] [--version] [-v] [--private-key PRIVATE_KEY_FILE] [-u REMOTE_USER]
                        [-c CONNECTION] [-T TIMEOUT] [--ssh-common-args SSH_COMMON_ARGS]
                        [--sftp-extra-args SFTP_EXTRA_ARGS] [--scp-extra-args SCP_EXTRA_ARGS]
                        [--ssh-extra-args SSH_EXTRA_ARGS]
                        [-k | --connection-password-file CONNECTION_PASSWORD_FILE]
                        [--force-handlers] [--flush-cache] [-b] [--become-method BECOME_METHOD]
                        [--become-user BECOME_USER]
                        [-K | --become-password-file BECOME_PASSWORD_FILE] [-t TAGS]
                        [--skip-tags SKIP_TAGS] [-C] [--syntax-check] [-D] [-i INVENTORY]
                        [--list-hosts] [-l SUBSET] [-e EXTRA_VARS] [--vault-id VAULT_IDS]
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [FirstPlay] ***************************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [Task1_Debug_Module] ******************************************************
ok: [172.31.7.129] => {
    "msg": "My Worker Node is executed"
}
ok: [172.31.8.49] => {
    "msg": "My Worker Node is executed"
}

PLAY RECAP *********************************************************************
172.31.7.129               : ok=2    changed=0    unreachable=0    failed=0    skipped
d=0    ignored=0
172.31.8.49                : ok=2    changed=0    unreachable=0    failed=0    skipped
d=0    ignored=0
```

2. Storing output of one module in the playbook via register function and printing it DEBUG module
   → Now we want to print the hostname of each worker node executing, when we run the playbook. So, In the playbook, We will use the command module with hostname-s command, to fetch the hostname of the worker node and the use register to define a variable name which will be called in the debug module.
   In the debug module, call that variable with .stdout function to print the output generated, from the variable which will be same as the output from the previous task. We will be able to successfully see the hostnames of both the worker nodes getting printed in the output window.
   Here, hostname –s value in the command module fetches the host names of the server || register stores the variable name of that module || hostname_output_name.stdout prints the output generated from that variable.
   please note* You cannot have space between "=" sign while defining any key value pair

ansiuser@AnsibleControllerMachine: ~

```
- name: FirstPlay With Variable
  hosts: webserver
  tasks:
  - name: command for fetching hostname of server
    command: hostname -s
    register: hostname_output_name
  - name: Task1 Debug Module print variable
    debug: var=hostname_output_name.stdout
~
~
~
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ vim First_Ansible_playbook.yml
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [FirstPlay With Variable] *****************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [command for fetching hostname of server] *************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [Task1 Debug Module print variable] *******************************************
ok: [172.31.7.129] => {
    "hostname_output_name.stdout": "AnsibleNode1"
}
ok: [172.31.8.49] => {
    "hostname_output_name.stdout": "AnsibleNode2"
}

PLAY RECAP *************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped
d=0    ignored=0
```

3. Using the Custom Variables

➔ Let us see the scenario where we try to define the git package and its state and install git on host servers. We can perform this task through variables.
➔ Syntax for defining the variables.

```
Variable in a playbook are written ina seprate section called as vars

vars:
 var1: value
 valr2: value

Call/refer the variable in tasks:

{{varname}}
```

```
ansiuser@AnsibleControllerMachine:~$ cat First_Ansible_playbook.yml
- name: Explore variables
  hosts: webserver
  vars:
   pkg_name: git
   pkg_status: present
  tasks:
  - name: my random module
    package: name={{pkg_name}} state={{pkg_status}}
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml -b

PLAY [Explore variables] ***********************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my random module] ************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

PLAY RECAP *************************************************************************
172.31.7.129               : ok=2    changed=0    unreachable=0    failed=0    skipped=
172.31.8.49                : ok=2    changed=0    unreachable=0    failed=0    skipped=
```

*please note- The apt module does the same work as package module with the same pair of arguments and values. The only difference is that the apt module is only applicable on the Linux distribution systems but package module is the generic module which can be used in any OS distribution

➔ More Examples of variables:

```
  ^ here
ansiuser@AnsibleControllerMachine:~$ vim First_Ansible_playbook.yml
ansiuser@AnsibleControllerMachine:~$ cat First_Ansible_playbook.yml
- name: Explore variables
  hosts: webserver
  vars:
   pkg_name: openjdk-8-jdk
   pkg_state: present
  tasks:
  - name: my random module
    apt: name:{{pkg_name}} state:{{pkg_state}}
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.ym
l

PLAY [Explore variables] ***********************************************
*

TASK [Gathering Facts] *************************************************
*
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my random module] ************************************************
*
ok: [172.31.7.129]
ok: [172.31.8.49]

PLAY RECAP *********************************************************************
172.31.7.129               : ok=2    changed=0    unreachable=0    failed=0    skipped=0
172.31.8.49                : ok=2    changed=0    unreachable=0    failed=0    skipped=0
```

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  vars:
   pkg_name: openjdk-8-jdk
   pkg_state: present
   file_path: /tmp/file1.txt
   file_state: touch
  tasks:
  - name: my apt module
    apt: name:{{pkg_name}} state:{{pkg_state}}
  - name: my file module
    file: path={{file_path}} state={{file_state}}
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ vim First_Ansible_playbook.yml
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************************

TASK [Gathering Facts] *************************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [my apt module] ***************************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my file module] **************************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

PLAY RECAP *************************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped=0    r
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipped=0    r

ansiuser@AnsibleControllerMachine:~$
```

➔ Sometimes the worker nodes needs the user to be the root user to run some modules like installation of some packages on the machine and so on. For that we will add the section of root user privilege in the playbook.
We can also define this in the ansible config file under the root user privilege section, so that we don't need to mention it in the playbook.

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  vars:
   pkg_name: git
   pkg_state: present
   file_path: /tmp/file1.txt
   file_state: touch
  tasks:
  - name: my apt module
    apt: name:{{pkg_name}} state:{{pkg_state}}
  - name: my file module
    file: path={{file_path}} state={{file_state}}
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ vim First_Ansible_playbook.yml
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my apt module] ***************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my file module] **************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

PLAY RECAP *************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    sk
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    sk

ansiuser@AnsibleControllerMachine:~$
```

➔ Place the variables in the separate file or multiple files and call the file name in the playbook section. If the variable files are present in the different directory than give the path for the variable file.
vars_files – is a attribute used to call the variable file in the playbook.

```
ansiuser@AnsibleControllerMachine: ~
pkg_name: git
pkg_state: present
file_path: /tmp/file1.txt
file_state: touch

~
~
~
```

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  vars_files:
   - my_variables.yml
  tasks:
  - name: my apt module
    apt: name:{{pkg_name}} state:{{pkg_state}}
  - name: my file module
    file: path={{file_path}} state={{file_state}}
~
~
~
~
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ vim First_Ansible_playbook.yml
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my apt module] ***************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my file module] **************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

PLAY RECAP *************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipp
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipp
```

➔ If a package is unable to get installed on the worker nodes because apt repository need to get updated , we update the apt repository by using command module in the task section to define the command- apt-get update

```
ansiuser@AnsibleControllerMachine: ~
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  vars_files:
   - my_variables.yml
  tasks:
  - name: update apt repository
    command: apt-get update
  - name: my apt module
    apt: name:{{pkg_name}} state:{{pkg_state}}
  - name: my file module
    file: path={{file_path}} state={{file_state}}
~
~
~
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [update apt repository] *******************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [my apt module] ***************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [my file module] **************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

PLAY RECAP *************************************************************************
172.31.7.129               : ok=4    changed=2    unreachable=0    failed=0    skipped=0
d=0
172.31.8.49                : ok=4    changed=2    unreachable=0    failed=0    skipped=0
d=0
```

*kindly note: sometimes the playbook does not execute, showing the syntax error for ":" key value pair, as shown in example. There we define arguments key values in the different line under the task module section.

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  tasks:
   - name: update apt repository
     command: apt-get update
   - name: install apache2
     package: name: apache2 state: present
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml
ERROR! We were unable to read either as JSON nor YAML, these are the errors we got
JSON: Expecting value: line 1 column 1 (char 0)

Syntax Error while loading YAML.
  mapping values are not allowed in this context

The error appears to be in '/home/ansiuser/First_Ansible_playbook.yml': line 9, co
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

   - name: install apache2
     package: name: apache2 state: present
                  ^ here
```

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  tasks:
   - name: update apt repository
     command: apt-get update
   - name: install apache2
     package:
      name: apache2
      state: present
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [update apt repository] *******************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [install apache2] *************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

PLAY RECAP *************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipp
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipp
```

➔ Defining the variables at runtime by the user using prompt function.
Here,
vars_prompt attribute specifies the runtime variables configurations and features.
Prompt section defines the statement you want to show to user for before entering the variable
Private defines if the entered variable by user should be visible, in hidden private format or not. For example, if you are defining the password to be entered by the user, then you set the private as true.

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  vars_prompt:
   - name: pkg_name
     prompt: Enter the Package name you want to install
     private: false
   - name: pkg_state
     prompt: Enter the state of package
     private: false
  tasks:
   - name: update apt repository
     command: apt-get update
   - name: my apt module
     apt: name:{{pkg_name}} state:{{pkg_state}}
     #- name: my file module
     #file: path={{file_path}} state={{file_state}}
~
~
~
~
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml
Enter the Package name you want to install: apache2
Enter the state of package: present

PLAY [Explore variables] ******************************************************************

TASK [Gathering Facts] ********************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [update apt repository] **************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [my apt module] **********************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

PLAY RECAP ********************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipped
```

4. Fact Variables & When condition

- ➔ Facts are the snippet of information about host server, stored in the fact variables. There are around 900 or above fact variable for each server. All the facts variables can be seen by the command #ansible <target_host> -m setup. Here it will be - #ansible webserver –m setup.
- ➔ You can also put filters on the commands to see only a particular section of information about the host server as seen in the screenshots.
- ➔ These are the variables who's values and names are given by the ansible. We cannot simply define them unlike the previous custom variables.
- ➔ These variables are used for dynamically updating the Configurations on the host servers.

ansiuser@AnsibleControllerMachine: ~

```
Some actions do not make sense in Ad-Hoc (include, meta, etc)
ansiuser@AnsibleControllerMachine:~$ ansible webserver -m setup
172.31.7.129 | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "172.31.7.129"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::8f6:e4ff:fe6a:eef0"
        ],
        "ansible_apparmor": {
            "status": "enabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "08/24/2006",
        "ansible_bios_vendor": "Xen",
        "ansible_bios_version": "4.11.amazon",
        "ansible_board_asset_tag": "NA",
        "ansible_board_name": "NA",
        "ansible_board_serial": "NA",
        "ansible_board_vendor": "NA",
        "ansible_board_version": "NA",
        "ansible_chassis_asset_tag": "NA",
        "ansible_chassis_serial": "NA",
        "ansible_chassis_vendor": "Xen",
        "ansible_chassis_version": "NA",
        "ansible_cmdline": {
            "BOOT_IMAGE": "/boot/vmlinuz-5.15.0-1036-aws",
            "console": "ttyS0",
            "nvme_core.io_timeout": "4294967295",
            "panic": "-1",
            "ro": true,
            "root": "PARTUUID=649d63cb-5b71-44b7-8495-2466cb37066f"
        },
        "ansible_date_time": {
            "date": "2023-10-22",
            "day": "22",
            "epoch": "1697952767",
            "epoch_int": "1697952767",
            "hour": "05",
```

```
}
ansiuser@AnsibleControllerMachine:~$ ansible webserver -m setup -a "filter=ansible_os*"
172.31.7.129 | SUCCESS => {
    "ansible_facts": {
        "ansible_os_family": "Debian",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
172.31.8.49 | SUCCESS => {
    "ansible_facts": {
        "ansible_os_family": "Debian",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
```

```
ansiuser@AnsibleControllerMachine:~$ ansible webserver -m setup -a "filter=ansible_default_ipv4*"
172.31.8.49 | SUCCESS => {
    "ansible_facts": {
        "ansible_default_ipv4": {
            "address": "172.31.8.49",
            "alias": "eth0",
            "broadcast": "172.31.15.255",
            "gateway": "172.31.0.1",
            "interface": "eth0",
            "macaddress": "0a:41:ca:0a:e4:52",
            "mtu": 9001,
            "netmask": "255.255.240.0",
            "network": "172.31.0.0",
            "type": "ether"
        },
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
172.31.7.129 | SUCCESS => {
    "ansible_facts": {
        "ansible_default_ipv4": {
            "address": "172.31.7.129",
            "alias": "eth0",
            "broadcast": "172.31.15.255",
            "gateway": "172.31.0.1",
            "interface": "eth0",
            "macaddress": "0a:f6:e4:6a:ee:f0",
            "mtu": 9001,
            "netmask": "255.255.240.0",
            "network": "172.31.0.0",
            "type": "ether"
        },
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

```
ansiuser@AnsibleControllerMachine:~$ ansible webserver -m setup -a "filter=ansible_system*"
172.31.7.129 | SUCCESS => {
    "ansible_facts": {
        "ansible_system": "Linux",
        "ansible_system_capabilities": [
            ""
        ],
        "ansible_system_capabilities_enforced": "True",
        "ansible_system_vendor": "Xen",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
172.31.8.49 | SUCCESS => {
    "ansible_facts": {
        "ansible_system": "Linux",
        "ansible_system_capabilities": [
            ""
        ],
        "ansible_system_capabilities_enforced": "True",
        "ansible_system_vendor": "Xen",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

➔ These fact variables like “ansible_os_family” , “ansible_default_ipv4”, etc… are used to defines the conditions in the playbook tasks to execute a specific task, if that condition is satisfied. We use When function for that.

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  tasks:
   - name: update apt repository
     command: apt-get update
   - name: install apache2
     package:
      name: apache2
      state: present
     when: ansible_os_family == "Debian"
   - name: install httpd
     package:
      name: httpd
      state: present
     when: ansible_os_family != "Debian"
~
~
~
~
~
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [update apt repository] *******************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [install apache2] *************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [install httpd] ***************************************************************
skipping: [172.31.7.129]
skipping: [172.31.8.49]

PLAY RECAP *************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipped
```

➔ Let us see another example of when condition

```
172.31.8.49
ansiuser@AnsibleControllerMachine:~$ ansible webserver -m setup
172.31.8.49 | SUCCESS => {
    "ansible_facts": {
```

```
            }
        },
        "ansible_distribution": "Ubuntu",
        "ansible_distribution_file_parsed": true,
        "ansible_distribution_file_path": "/etc/os-release",
        "ansible_distribution_file_variety": "Debian",
        "ansible_distribution_major_version": "20",
        "ansible_distribution_release": "focal",
        "ansible_distribution_version": "20.04",
```

ansiuser@AnsibleControllerMachine: ~

```
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  tasks:
   - name: update apt repository
     command: apt-get update
   - name: install apache2
     package:
      name: apache2
      state: present
     when: (ansible_distribution == "CentOS" and ansible_distribution_major_version == "18") or
           (ansible_distribution == "Ubuntu" and ansible_distribution_major_version == "20")
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ vim First_Ansible_playbook.yml
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************************

TASK [Gathering Facts] *************************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [update apt repository] *******************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [install apache2] *************************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

PLAY RECAP *************************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=
```

Let us use register function with when condition

```
ansiuser@AnsibleControllerMachine: ~
- name: Explore variables
  hosts: webserver
  become: true
  become_user: root
  tasks:
          - name: Checking if the host server satisfies all consitions with register function
            command: hostname -s
            when: (ansible_distribution == "CentOS" and ansible_distribution_major_version == "18") or
                  (ansible_distribution == "Ubuntu" and ansible_distribution_major_version == "20")
            register: my_variable
          - name: Print the hostname satisfying the condition
            debug: var= my_variable.stdout
```

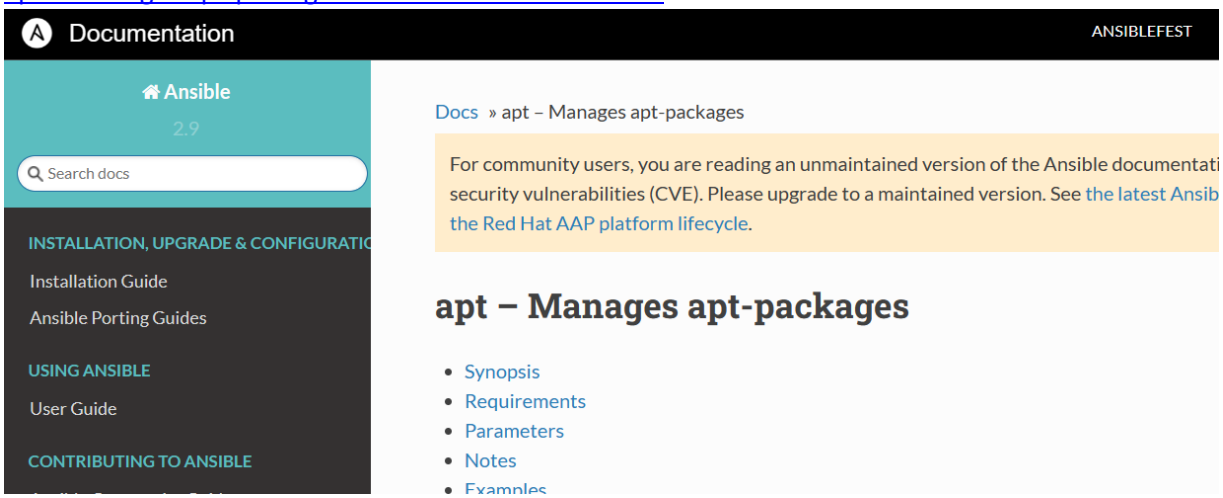
```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore variables] ***********************************************************

TASK [Gathering Facts] *************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [Checking if the host server satisfies all consitions with register function] ********
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [Print the hostname satisfying the condition] *********************************
ok: [172.31.7.129] => {
    "my_variable.stdout": "AnsibleNode1"
}
ok: [172.31.8.49] => {
    "my_variable.stdout": "AnsibleNode2"
}

PLAY RECAP *************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped=0
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipped=0
```

5. LOOPS

➔ When we need to define the same module multiple times with different values, we use loops in playbook.
Let's take a scenario where we will try to install multiple packages at the same time in our host servers.

We will use apt module this time to update the cache files, in place of command module with apt get update function.
We will enclose item variable in the "double" quote, because it is the pre-defined variable and will be considered as string if we do not enclose it.

[apt – Manages apt-packages — Ansible Documentation]



	• present ← • fixed	
update_cache boolean	Choices: • no ← • yes	Run the equivalent of `apt-get update` before the operation. Can be run as part of the package installation or as a separate step.
upgrade	Choices:	If yes or safe, performs an aptitude safe-upgrade.

		See documentation for further information.
cache_valid_time -	Default: 0	Update the apt cache if its older than the *cache_valid_time*. This option is set in seconds. As of Ansible 2.4, if explicitly set, this sets *update_cache=yes*.
deb		Path to a .deb package on the remote machine.

ansiuser@AnsibleControllerMachine: ~

```
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  tasks:
  - name: update apt cache
    apt:
     update_cache: yes
     cache_valid_time: 3600
  - name: install all the packages at once
    apt:
     name: "{{item}}"
     state: present
    loop:
    - git
    - apache2
~
~
~
~
~
~
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore loops] ***********************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [update apt cache] ********************************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [install all the packages at once] ****************************************************************
ok: [172.31.7.129] => (item=git)
ok: [172.31.8.49] => (item=git)
ok: [172.31.7.129] => (item=apache2)
ok: [172.31.8.49] => (item=apache2)

PLAY RECAP *********************************************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescue
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescue
```

➔ Creating multiple users with help of variables user list and calling that variable in the loop function

ansiuser@AnsibleControllerMachine: ~

```
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   User_list:
   - User_1
   - User_2
   - User_3
   - User_4
  tasks:
  - name: update apt cache
    apt:
     update_cache: yes
     cache_valid_time: 3600
  - name: install all the packages at once
    user:
     name: "{{item}}"
     state: present
    loop: "{{User_list}}"
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ vim First_Ansible_playbook.yml
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore loops] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [update apt cache] ********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [install all the packages at once] ****************************************
changed: [172.31.7.129] => (item=User_1)
changed: [172.31.8.49] => (item=User_1)
changed: [172.31.7.129] => (item=User_2)
changed: [172.31.8.49] => (item=User_2)
changed: [172.31.8.49] => (item=User_3)
changed: [172.31.7.129] => (item=User_3)
changed: [172.31.8.49] => (item=User_4)
changed: [172.31.7.129] => (item=User_4)

PLAY RECAP *********************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipped=0
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipped=0
```

6. Tags

➔ When our playbook is consisting of multiple tasks to be executed, we set tags in each task, which can be called while executing the playbook, on basis of which the task will be executed accordingly. It is used to create a filter in execution of tasks.

➔ Whatever tags we provide in the execute command. Only tasks containing those tags will be executed.

ansiuser@AnsibleControllerMachine: ~

```
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   User_list:
   - User_1
   - User_2
   - User_3
   - User_4
  tasks:
  - name: create all users at once
    user:
     name: "{{item}}"
     state: present
    loop: "{{User_list}}"
  - name: execute command
    command: hostname -s
    tags: command
  - name: execute command 2
    command: echo "Hello ALL"
    tags: command
  - name: install package
    package:
     name: tree
     state: present
    tags: install
  - name: uninstall package
    package:
     name: vim
     state: absent
    tags: uninstall
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml --tags instal
l

PLAY [Explore loops] ***********************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [install package] *********************************************************************
changed: [172.31.7.129]
changed: [172.31.8.49]

PLAY RECAP *********************************************************************************
172.31.7.129               : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescue
172.31.8.49                : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescue
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml --tags uninstall

PLAY [Explore loops] ***********************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [uninstall package] *******************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

PLAY RECAP *********************************************************************************
172.31.7.129               : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
172.31.8.49                : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml --tags command

PLAY [Explore loops] ***********************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [execute command] *********************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [execute command 2] *******************************************************************
changed: [172.31.7.129]
changed: [172.31.8.49]

PLAY RECAP *********************************************************************************
172.31.7.129               : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued
172.31.8.49                : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued
```

Untagged tasks – task which have no tags present

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml --tags untagged

PLAY [Explore loops] ***********************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [create all users at once] ************************************************************************
ok: [172.31.7.129] => (item=User_1)
ok: [172.31.8.49] => (item=User_1)
ok: [172.31.7.129] => (item=User_2)
ok: [172.31.8.49] => (item=User_2)
ok: [172.31.7.129] => (item=User_3)
ok: [172.31.8.49] => (item=User_3)
ok: [172.31.7.129] => (item=User_4)
ok: [172.31.8.49] => (item=User_4)

PLAY RECAP *********************************************************************************************
172.31.7.129               : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescue
172.31.8.49                : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescue
```

➔ Deploying HTML file on the host servers
We will create a html file in the same directory and use copy module to copy the file on host servers

```yaml
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   pkg_name: vim

  tasks:
  - name: install package
    package:
     name: "{{pkg_name}}"
     state: present
    when: ansible_distribution == "Ubuntu"
    tags: install
  - name: uninstall package
    package:
     name: "{{pkg_name}}"
     state: absent
    tags: uninstall
    when: ansible_distribution != "Ubuntu"
  - name: Deploy HTML File
    copy:
     src: myHtmlFile.html
     dest: /home/ansiuser/html
~
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore loops] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [install package] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [uninstall package] *******************************************************
skipping: [172.31.7.129]
skipping: [172.31.8.49]

TASK [Deploy HTML File] ********************************************************
changed: [172.31.7.129]
changed: [172.31.8.49]

PLAY RECAP *********************************************************************
172.31.7.129               : ok=3    changed=1    unreachable=0    failed=0    skipp
 ignored=0
172.31.8.49                : ok=3    changed=1    unreachable=0    failed=0    skipp
 ignored=0
```

7. Handlers

➔ We will see a small example for understanding this function.

```
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   pkg_name: vim

  tasks:
  - name: install package
    package:
     name: "{{pkg_name}}"
     state: present
    when: ansible_distribution == "Ubuntu"
    tags: install
  - name: uninstall package
    package:
     name: "{{pkg_name}}"
     state: absent
    tags: uninstall
    when: ansible_distribution != "Ubuntu"
  - name: start the apache2 server
    service:
     name: apache2
     state: started
  - name: Deploy HTML File
    copy:
     src: myHtmlFile.html
     dest: /home/ansiuser/html
  - name: restart the apache2 server
    service:
     name: apache2
     state: restarted
~
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore loops] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [install package] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [uninstall package] *******************************************************
skipping: [172.31.7.129]
skipping: [172.31.8.49]

TASK [start the apache2 server] ************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [Deploy HTML File] ********************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [restart the apache2 server] **********************************************
changed: [172.31.7.129]
changed: [172.31.8.49]

PLAY RECAP *********************************************************************
172.31.7.129               : ok=5    changed=1    unreachable=0    failed=0    skip
 ignored=0
172.31.8.49                : ok=5    changed=1    unreachable=0    failed=0    skip
 ignored=0
```

As we can see here the task of deployment of HTML file has status as OK. Not CHANGED. Because the file was already deployed and present on the server. So the task with restarting the apache 2 server does not contribute in this case. Hence, we can set the dependency of task of restarting apache server, based on the status of previous task.

We can do that via handlers

➔ We have set here that , if the deployment tasks is executed and status is changed, the ansible will notify both the tasks mentioned in the notify section to be executed present in the handlers section.
We can see here, as the deployment task status was not changed, the notify function did not notify the tasks, hence the tasks under the handlers section did not execute. This saves us the plenty of unwanted steps and hence reduces the execution time in the big projects.

```
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   pkg_name: vim

  tasks:
  - name: install package
    package:
     name: "{{pkg_name}}"
     state: present
    when: ansible_distribution == "Ubuntu"
    tags: install
  - name: uninstall package
    package:
     name: "{{pkg_name}}"
     state: absent
    tags: uninstall
    when: ansible_distribution != "Ubuntu"
  - name: start the apache2 server
    service:
     name: apache2
     state: started
  - name: Deploy HTML File
    copy:
     src: myHtmlFile.html
     dest: /home/ansiuser/html
    notify:
    - restart the apache2 server
    - status of deployment

  handlers:
  - name: restart the apache2 server
    service:
     name: apache2
     state: restarted
  - name: status of deployment
    debug:
     msg: "The deployment is successful"
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore loops] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [install package] *********************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [uninstall package] *******************************************************
skipping: [172.31.7.129]
skipping: [172.31.8.49]

TASK [start the apache2 server] ************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [Deploy HTML File] ********************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

PLAY RECAP *********************************************************************
172.31.7.129               : ok=4    changed=0    unreachable=0    failed=0    skippe
 ignored=0
172.31.8.49                : ok=4    changed=0    unreachable=0    failed=0    skippe
 ignored=0
```

➔ Now we have added a new html file in our playbook to be deployed onto the host servers. Hence, the deployment task has changed its status to changed, and notified the handlers section task, to be executed successfully.

```
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   pkg_name: vim

  tasks:
  - name: install package
    package:
     name: "{{pkg_name}}"
     state: present
    when: ansible_distribution == "Ubuntu"
    tags: install
  - name: uninstall package
    package:
     name: "{{pkg_name}}"
     state: absent
    tags: uninstall
    when: ansible_distribution != "Ubuntu"
  - name: start the apache2 server
    service:
     name: apache2
     state: started
  - name: Deploy HTML File
    copy:
     src: new_html.html
     dest: /home/ansiuser/html
    notify:
    - restart the apache2 server
    - status of deployment

  handlers:
  - name: restart the apache2 server
    service:
     name: apache2
     state: restarted
  - name: status of deployment
    debug:
     msg: "The deployment is successful"
~
~
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore loops] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [install package] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [uninstall package] *******************************************************
skipping: [172.31.7.129]
skipping: [172.31.8.49]

TASK [start the apache2 server] ************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [Deploy HTML File] ********************************************************
changed: [172.31.7.129]
changed: [172.31.8.49]

RUNNING HANDLER [restart the apache2 server] ***********************************
changed: [172.31.7.129]
changed: [172.31.8.49]

RUNNING HANDLER [status of deployment] *****************************************
ok: [172.31.7.129] => {
    "msg": "The deployment is successful"
}
ok: [172.31.8.49] => {
    "msg": "The deployment is successful"
}

PLAY RECAP *********************************************************************
172.31.7.129               : ok=6    changed=2    unreachable=0    failed=0    skipp
 ignored=0
172.31.8.49                : ok=6    changed=2    unreachable=0    failed=0    skipp
 ignored=0
```

➔ Handlers tasks are always executed at the end of all the tasks, even if they are notified prior to other tasks. Notif function only occurs when the task status is- "changed"
Here we can see, the 1st task was executed and status was changed hence, it notified the handler user creation task. But in the execution process, the user creation task was executed at the last. This shows that handlers tasks are executed at the end.

```
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   user_name: MyUser

  tasks:
  - name: install package
    user:
     name: "{{user_name}}"
     state: present
    notify:
    - status of user creation
  - name: start the apache2 server
    service:
     name: apache2
     state: started
  - name: Deploy HTML File
    copy:
     src: new_html.html
     dest: /home/ansiuser/html
    notify:
    - restart the apache2 server
    - status of deployment

  handlers:
  - name: restart the apache2 server
    service:
     name: apache2
     state: restarted
  - name: status of deployment
    debug:
     msg: "The deployment is successful"
  - name: status of user creation
    debug:
     msg: "The creation of user is successful"

~
```

Please note* The 2nd task named- install package here is actually the User creation task. I forgot to rename it.

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.ym

PLAY [Explore loops] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [install package] *********************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [start the apache2 server] ************************************************
ok: [172.31.8.49]
ok: [172.31.7.129]

TASK [Deploy HTML File] ********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

RUNNING HANDLER [status of user creation] **************************************
ok: [172.31.8.49] => {
    "msg": "The creation of user is successful"
}
ok: [172.31.7.129] => {
    "msg": "The creation of user is successful"
}

PLAY RECAP *********************************************************************
172.31.7.129               : ok=5    changed=1    unreachable=0    failed=0
 ignored=0
172.31.8.49                : ok=5    changed=1    unreachable=0    failed=0
 ignored=0
```

- ➔ But if we want to execute the handler task, just after it is notified, then we use the flush function.
  Here, we use meta module to define that the handlers that should run at this step of execution if notified.
  Please note* The flush function will only be applicable for the tasks above this function.
- ➔ As we can see, the handlers tasks were executed just after they are notified in the user creation task.

```yaml
---
- name: Explore loops
  hosts: webserver
  become: true
  become_user: root
  vars:
   user_name: MyUser2

  tasks:
  - name: create user
    user:
     name: "{{user_name}}"
     state: present
    notify:
    - status of user creation
  - name: Flush the handlers. makes sure they are executed here
    meta: flush_handlers

  - name: start the apache2 server
    service:
     name: apache2
     state: started
  - name: Deploy HTML File
    copy:
     src: new_html.html
     dest: /home/ansiuser/html
    notify:
    - restart the apache2 server
    - status of deployment

  handlers:
  - name: restart the apache2 server
    service:
     name: apache2
     state: restarted
  - name: status of deployment
    debug:
     msg: "The deployment is successful"
  - name: status of user creation
    debug:
     msg: "The creation of user is successful"
```

```
ansiuser@AnsibleControllerMachine:~$ ansible-playbook First_Ansible_playbook.yml

PLAY [Explore loops] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [create user] *************************************************************
changed: [172.31.8.49]
changed: [172.31.7.129]

TASK [Flush the handlers. makes sure they are executed here] *******************

RUNNING HANDLER [status of user creation] **************************************
ok: [172.31.8.49] => {
    "msg": "The creation of user is successful"
}
ok: [172.31.7.129] => {
    "msg": "The creation of user is successful"
}

TASK [start the apache2 server] ************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

TASK [Deploy HTML File] ********************************************************
ok: [172.31.7.129]
ok: [172.31.8.49]

PLAY RECAP *********************************************************************
172.31.7.129               : ok=5    changed=1    unreachable=0    failed=0    skipp
 ignored=0
172.31.8.49                : ok=5    changed=1    unreachable=0    failed=0    skipp
 ignored=0
```

➔ Fail module: This module fails the whole playbook even if every task was executed and changed. Hence, even the task status was changed and the notification was executed, the handler section will not be executed because the playbook was failed.

```
tasks:
- name: Install {{pkg_name}}
  package: name={{pkg_name}} state=present
  when: ansible_distribution == "Ubuntu"
- name: User creation
  user: name=deploy2 state=present
  notify: User creation success
- name: Start apache2 server
  service: name={{pkg_name}} state=started
- name: deploy index.html file
  copy: src=index.html dest=/var/www/html
  notify:
   - restart the server apache2
   - Deployment status
- name: Failure task  # this fail module will fail your playbook
  fail:
handlers: # if any parent task status is changed, that task will notify the handler task to execute
- name: restart the server apache2
  service: name={{pkg_name}} state=restarted
- name: Deployment status
  debug: msg="Deployment successfull"
```

```
TASK [deploy index.html file] ******************************************************************
changed: [172.31.47.98]
changed: [172.31.42.13]

TASK [Failure task] ****************************************************************************
fatal: [172.31.47.98]: FAILED! => {"changed": false, "msg": "Failed as requested from task"}
fatal: [172.31.42.13]: FAILED! => {"changed": false, "msg": "Failed as requested from task"}

RUNNING HANDLER [restart the server apache2] ***************************************************

RUNNING HANDLER [Deployment status] ************************************************************

RUNNING HANDLER [User creation success] ********************************************************

PLAY RECAP *************************************************************************************
172.31.42.13               : ok=5    changed=2    unreachable=0    failed=1    skipped=0    rescued=0    igno
ed=0
172.31.47.98               : ok=5    changed=2    unreachable=0    failed=1    skipped=0    rescued=0    igno
ed=0
```

➔ Force handlers: specifying the force handers in the playbook execution command, defines handlers to get executed, even if the playbook fails. It is just used in contrary to the fail module.
Here, on the same playbook, when executed with the force handlers command, it executed the handlers section tasks successfully.

```
ansiuser@ip-172-31-23-45:~$ ansible-playbook deployplaybook.yml --force-handlers

PLAY [deploy html page on apache2] *********************************************

TASK [Gathering Facts] *********************************************************
ok: [172.31.47.98]
ok: [172.31.42.13]

TASK [Install apache2] *********************************************************
ok: [172.31.47.98]
ok: [172.31.42.13]

TASK [User creation] ***********************************************************
changed: [172.31.47.98]
changed: [172.31.42.13]

TASK [Start apache2 server] ****************************************************
ok: [172.31.42.13]
ok: [172.31.47.98]

TASK [deploy index.html file] **************************************************
```

```
TASK [Failure task] ************************************************************************************
fatal: [172.31.47.98]: FAILED! => {"changed": false, "msg": "Failed as requested from task"}
fatal: [172.31.42.13]: FAILED! => {"changed": false, "msg": "Failed as requested from task"}

RUNNING HANDLER [User creation success] ****************************************************************
ok: [172.31.47.98] => {
    "msg": "User creation is  successfull"
}
ok: [172.31.42.13] => {
    "msg": "User creation is  successfull"
}

PLAY RECAP *********************************************************************************************
172.31.42.13               : ok=6    changed=1    unreachable=0    failed=1    skipped=0    rescued=0    ignor
ed=0
172.31.47.98               : ok=6    changed=1    unreachable=0    failed=1    skipped=0    rescued=0    ignor
ed=0
```