

Ansible Conditional Tasks

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Session - 17 Agenda:

1. Ansible Conditional Tasks (Ansible Conditionals)
-

Ansible Conditionals:

In Ansible, you can define conditions that will be evaluated before a task is executed. When a condition is not met, the task is then skipped. This is done with the **when** keyword, which accepts expressions that are typically based on a variable or a fact.

In a playbook, you may want to execute different tasks, or have different goals, depending on the value of a fact (data about the remote system), a variable, or the result of a previous task. You may want the value of some variables to depend on the value of other variables. Or you may want to create additional groups of hosts based on whether the hosts match other criteria. You can do all of these things with conditionals.

Each task can have its own conditions and there may be n number of such tasks present in an ansible playbook.

Syntax:

when: statement operator value

Example:

when: ansible_os_family == 'RedHat'

Statement Value is called from the variables. It may be a:

- (a) *custom variable* like *command line variable*, *playbook variable* or *inventory variable*.
- (b) *setup variable*.
- (c) *registered variable*.

Operators:

(a) *comparison operators*: These are used once in a task for the comparison.

- `==` equals to
- `!=` not equal to
- `=>` greater than or equal to
- `<=` less than or equal to
- `>` greater than
- `<` less than

(b) *logical operators*: These are used for more than one statement in the same task.

AND for both statement values must be true.

OR any of the statement values must be true

If the condition is false, the task is skipped not cancelled. Ansible moves to next task in the playbook in such a case.

Let's understand ansible conditionals with some real time-based examples.



Ansible Conditional Tasks

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Using Ansible facts as statement values in ansible conditionals.

Let's create an ansible playbook to check the OS flavor of the managed node and print the output on the terminal.

```
$ vim ansible_conditionals.yml
```

```
---
```

```
- name: Ansible OS check playbook.  
hosts: all  
tasks:  
  - name: Checking whether OS is Redhat and print a msg.  
    shell: echo "This is a RHEL based machine."  
    when: ansible_distribution == "RedHat"  
  
  - name: Checking whether OS is Ubuntu and print a msg.  
    shell: echo "This is an Ubuntu based machine."  
    when: ansible_distribution == "Ubuntu"  
...
```

Check the playbook for the errors.

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

Now, we can execute the ansible playbook to test the OS flavor of the managed nodes.

```
$ ansible-playbook ansible_conditionals.yml -v
```

Here we are using facts as statement values.

```
$ ansible all -m setup -a 'filter=ansible_distribution'
```

Let's create another playbook to check the OS flavor and install the apache web server in the managed nodes if it is a linux machine.

```
$ vim ansible_conditionals_apache.yml
```

```
- name: Example of conditionals in Ansible  
hosts: all  
become: true  
tasks:  
  - name: Install the apache web server on the RedHat based machines.  
    yum:  
      name: httpd  
      state: installed  
    when: ansible_distribution == 'RedHat'  
  - name: Install the apache web server on the Ubuntu based machines.  
    apt:  
      name: apache2  
      state: present  
    when: ansible_distribution == 'Ubuntu'  
...
```



Ansible Conditional Tasks

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Check the playbook for the errors.

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

Now, we can execute the ansible playbook.

```
$ ansible-playbook ansible_conditionals_apache.yml -v
```

Verify the same using ansible ad-hoc commands.

```
$ ansible node1 -m command -a 'rpm -qi httpd'  
$ ansible node3 -m command -a 'rpm -qi httpd'  
$ ansible node2 -m command -a 'apt list installed apache2'
```

Let's create another playbook to check the OS flavor and install the ftp package in the managed nodes but this time there will be more than one condition (we will use logical operators) in the same task e.g. only install the ftp package if the OS is RedHat and having major version as 7.

```
$ vim ansible_conditionals_apache2.yml
```

```
---
```

```
- name: Example of conditionals in Ansible
  hosts: all
  become: true
  tasks:
    - name: Install FTP package if the node is RedHat based machines & having major version 7.
      yum:
        name: ftp
        state: installed
        when: ansible_distribution == 'RedHat' and ansible_distribution_major_version == '7'
    # this condition can be presented like this using the lists under when.
    - name: Install FTP package if the node is RedHat based machines & having major version 7.
      yum:
        name: ftp
        state: installed
        when:
          - ansible_distribution == 'RedHat'
          - ansible_distribution_major_version == '7'
    # Similarly, only install the FTP package if the OS is RedHat or centOS.
    - name: Install the FTP package if the node OS is Redhat or CentOS based machines.
      yum:
        name: ftp
        state: installed
        when: ansible_distribution == 'RedHat' or 'centos'
  ...
```

Check the playbook for the errors.

```
$ ansible-playbook ansible_conditionals_apache2.yml --syntax-check
```

Now, we can execute the ansible playbook.

```
$ ansible-playbook ansible_conditionals_apache2.yml
```



Ansible Conditional Tasks

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Verify the same using ansible ad-hoc commands.

```
$ ansible node1 -m command -a 'rpm -qi ftp'  
$ ansible node3 -m command -a 'rpm -qi ftp'
```

We can also merge multiple conditions in the same task using combination of logical operators. Suppose if we want to install the apache web server on the RHEL based machine and having major version 9 or CentOS based machines having major version 8.

```
$ vim ansible_conditionals_apache3.yml
```

```
---
```

```
- name: Example of conditionals in Ansible  
hosts: all  
become: true  
tasks:  
  - name: Install the apache web server on the RHEL based machine and having major  
version 9 or CentOS based machines having major version 8.  
    yum:  
      name: httpd  
      state: installed  
      when: (ansible_distribution == 'RedHat' and ansible_distribution_major_version ==  
'9') or (ansible_distribution == 'centos' and ansible_distribution_major_version == '8')  
...
```

Check the playbook for the errors.

```
$ ansible-playbook ansible_conditionals_apache3.yml --syntax-check
```

Now, we can execute the ansible playbook.

```
$ ansible-playbook ansible_conditionals_apache3.yml -v
```

Using registered variables (values) as statement values in ansible conditionals.

In the above examples, we were using ansible facts values as statement values which are gathered at the time of executing the playbook from the managed nodes. Now, we will stop the facts gathering so that we can use registered values as statements in ansible conditionals. Let's suppose we want to install vsftpd package in all the RHEL 9.0 machines only.

Let's verify that the vsftpd package is not already installed in the managed nodes.

```
$ ansible RHEL -m command -a 'rpm -qi vsftpd'
```

Let's create an ansible playbook to check the OS version and install the vsftpd package in the managed nodes which are running on RHEL 9.0.

Let's set the variable first.

```
$ vim ansible_conditionals_vsftpd.yml
```

```
---
```

```
- name: Ansible playbook to check the OS version and install the vsftpd package.  
hosts: all
```



Ansible Conditional Tasks

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
gather_facts: false
become: true
tasks:
  - name: Checking the machine OS version
    command: cat /etc/redhat-release
    register: os_version
  - debug:
      var: os_version
...
...
```

Check the playbook for the errors.

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

Now, we can execute the ansible playbook.

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

Let's complete the playbook and mention the task to install the vsftpd packages.

```
$ vim ansible_conditionals_vsftpd.yml
```

```
---
- name: Ansible playbook to check the OS version and install the vsftpd package.
hosts: all
gather_facts: false
become: true
tasks:
  - name: Checking the machine OS version
    command: cat /etc/redhat-release
    register: os_version
  - debug:
      var: os_version
  - name: Installing the VSFTP package if the OS version is RHEL 9.
    yum:
      name: vsftpd
      state: installed
    when: os_version.stdout == 'Red Hat Enterprise Linux release 9.0 (Plow)'

...
...
```

Check the playbook for the errors.

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

Now, we can execute the ansible playbook.

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

Let's verify that the vsftpd package is not already installed in the managed nodes.

```
$ ansible RHEL -m command -a 'rpm -qi vsftpd'
```

Let's take one more example, suppose if we want to install the zsh package in the



Ansible Conditional Tasks

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

managed nodes if and only if it has RAM more than 2GB.

```
$ vim ansible_conditionals_zsh.yml
```

```
- name: Example of conditionals in Ansible
hosts: RHEL
become: true
tasks:
  - name: Install the zsh package if the memory (RAM) is more than 2 GB.
    yum:
      name: zsh
      state: installed
    when: ansible_memtotal_mb > 2048
...
...
```

Let's verify that the vsftp package is not already installed in the managed nodes.

```
$ ansible RHEL -m command -a 'rpm -qi zsh'
```

Check the playbook for the errors.

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

Now, we can execute the ansible playbook.

```
$ ansible-playbook ansible_conditionals_zsh.yml -v
```

Let's verify that the vsftp package is not already installed in the managed nodes.

```
$ ansible RHEL -m command -a 'rpm -qi zsh'
```

Let's consider one more example here, suppose we want to check the existence of a file in the managed node and create one if not already available there.

```
$ vim ansible_conditionals_file.yml
```

```
- name: Example of conditionals in Ansible
hosts: all
become: true
tasks:
  - name: Checking the existence of the file.
    stat:
      path: /tmp/nehraclasses.txt
    register: res
  - debug:
      var: res.stat.exists
  - name: Creating the file if doesn't exists.
    copy:
      content: "Nehra Classes Are Awesome."
      dest: /tmp/nehraclasses.txt
    when: res.stat.exists == false
...
...
```



Ansible Conditional Tasks

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Check the playbook for the errors.

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

Now, we can execute the ansible playbook.

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

Working with Ansible when and loops.

You previously executed Ansible tasks according to an Ansible when for a particular parameter. But perhaps you need to check a condition with multiple parameters defined in a list. If so, then try adding a loop in a task.

The task in the code below (Task-1) runs a loop where the when condition checks if the item value is greater than five and returns the result.

```
$ vim ansible_conditionals_loop.yml
```

```
---
```

```
- name: Ansible tasks to work on Ansible When
  hosts: node1
  become: true
  tasks:
    # (Task -1) Checking if item value is greater than 5
    - name: Run with items greater than 5
      ansible.builtin.command: echo {{ item }}
      loop: [ 0, 2, 4, 6, 8, 10 ]
      when: item > 5
...
```

Check the playbook for the errors.

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

Now, we can execute the ansible playbook.

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

Thanks
