

Ansible Basic

An Ansible Training Course

PRESENTED BY:

Bittnet DevOps Team

ANSIBLE

We Make Custom Trainings



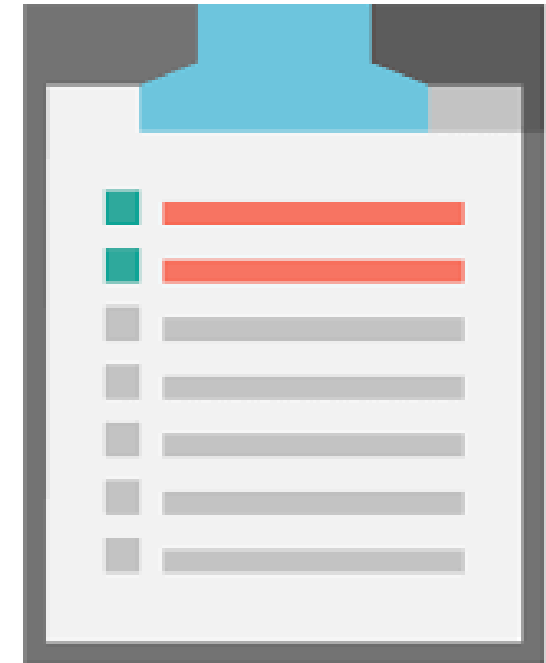
FASTER
SMARTER
SAFER

5. Facts, Variables



Topics covered:

- Facts - definition
- Fact gathering
- Disabling fact gathering
- Custom facts
- Variable definition



What are Facts?

- Ansible Facts are variables that are automatically set and discovered by Ansible on managed hosts.
- Facts contain information about hosts that can be used in conditionals.
- For instance, before installing specific software you can check that a managed host runs a specific kernel version.

Managing Fact Gathering

- By default, all playbooks perform fact gathering before running the actual plays
- You can run fact gathering manually by using the **setup** module
- To show facts, use the debug module to print the value of the **ansible_facts** variable

Displaying Fact Names

- In Ansible 2.4 and before, Ansible facts were stored as individual variables, such as **ansible_hostname** and **ansible_interfaces**.
- In Ansible 2.5 and later, all facts are stored in one variable with the name **ansible_facts**, and referring to specific facts happens in a different way:
 - **ansible_facts['hostname']**
 - **ansible_facts['interfaces']**
 - ...

Displaying Fact Names

```
student:~$ ansible all -m setup -a "filter=*ipv4"
hivemaster | SUCCESS => {
  "ansible_facts": {
    "ansible_default_ipv4": {
      "address": "10.128.0.48",
      "alias": "ens4",
      "broadcast": "global",
      "gateway": "10.128.0.1",
      "interface": "ens4",
      "macaddress": "42:01:0a:80:00:30",
      "mtu": 1460,
      "netmask": "255.255.255.255",
      "network": "10.128.0.48",
      "type": "ether"
    },
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false
}
[...]
```


Turning Off Fact Gathering

- Disabling fact gathering can significantly speed up playbook execution
- Use `gather_facts: no` in the play header to disable.
- If you need to use facts, you can collect them manually by running the `setup` module in a task.

```
- name: Disable facts
  hosts: all
  become: yes
  gather_facts: no
  tasks:
    - name: Print message
      debug:
        msg: "Fact gathering is disabled. Playbook is running faster"
```


Using Custom Facts

- Custom facts allow administrators to dynamically generate variables which are stored as facts
- Custom facts are stored in an ini or json file in the `/etc/ansible/facts.d` directory on the managed host
 - The name of these files must end in `.fact`
- Custom facts are stored in the **`ansible_facts.ansible_local`** variable
- Use **`ansible hostname -m setup -a "filter=ansible_local"`** to display local facts

Using Custom Facts

- Custom Facts Example File

```
[packages]
web_package = httpd
db_package = mariadb-server
[users]
user1 = joe
user2 = jane
```

```
[student ~]$ ansible demo1.example.com -m setup -a
'filter=ansible_local'
demo1.lab.example.com | SUCCESS => {
  "ansible_facts": {
    "ansible_local": {
      "custom": {
        "packages": {
          "db_package": "mariadb-server",
          "web_package": "httpd"
        },
        "users": {
          "user1": "joe",
          "user2": "jane"
        }
      }
    }
  },
  "changed": false
}
```

Variables

- A variable is a label that is assigned to a specific value to make it easy to refer to that value throughout the playbook
- Variables can be defined by administrators at different levels
- Variables are particularly useful when dealing with managed hosts where specifics are different
 - Set a variable `web_service` on Ubuntu and Red Hat
 - Refer to that variable `web_service` instead of the specific service name

Variables Definition

- Variables can be set at different levels
 - In a playbook
 - In the inventory file (not recommended for large numbers of variables)
 - In specific variable files
- Variable names have some requirements
 - The name must start with a letter
 - Variable names can only contain letters, numbers, and underscores

Variables Definition

- Variables can be defined in a vars block in the beginning of a playbook

```
- hosts: all
  vars:
    web_package: httpd
```

- Alternatively, variables can be defined in a variable file, which will be included from the playbook

```
- hosts: all
  vars_files:
    - vars/users.yml
```

Variables Definition

- After defining the variables, they can be used later in the playbook
- Referring to a variable:
 - `{{ web_package }}`
- If the variable is the first element, using quotes is mandatory!

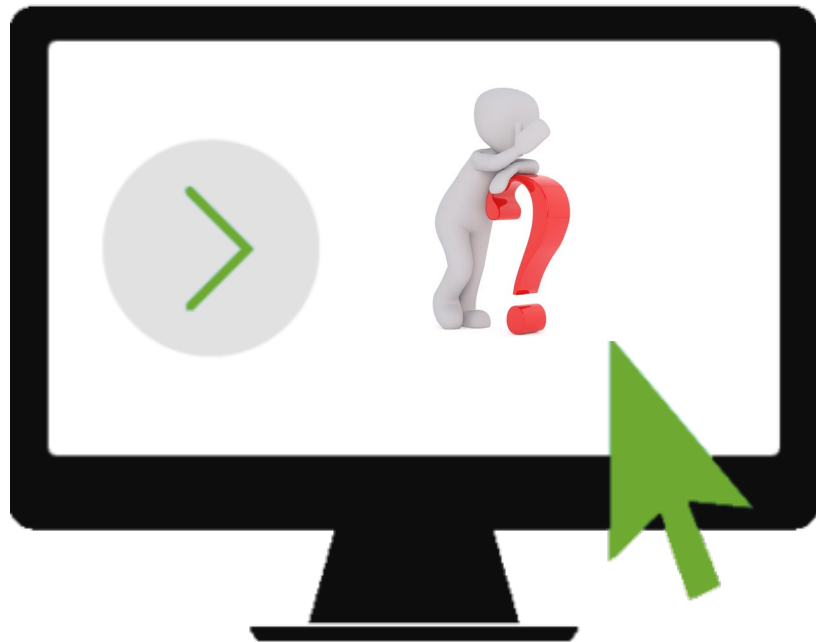
```
---  
- name: create a user using a variable  
  hosts: all  
  vars:  
    user: ben  
  tasks:  
    - name: create a user {{ user }}  
      user:  
        name: "{{ user }}"
```

Variables Definition

- Variables can be set with different types of scope
 - **Global scope:** set from config, envvars, or command line
 - **Play scope:** set from the playbook (and included files)
 - **Host scope:** set from the inventory, collected facts, or registered task outputs
- When the same variable is set at different levels, the most specific level gets precedence
- When a variable is set from the command line, it will overwrite anything else
 - IF set with -e "var=value"!

Variables Definition

- Some variables are build in and cannot be used for anything else
 - `hostvars`
 - `inventory_hostname`
 - `inventory_hostname_short`
 - `groups`
 - `group_names`
 - `ansible_check_mode`
 - `ansible_play_batch`
 - `ansible_play_hosts`
 - `ansible_version`



Lab 5: Facts, Variables





Solutions for training the world.