

Variable nə üçündür?

Variable daha çox iş mühitində lazım olur. Məsələn yeni bir playbook yazırıq, yum modulundan istifadə edib paket yükləyirik, daha sonra onu firewall`a əlavə edirik. Bunu fərqli paketlər üçün etdikdə məcbur oluruq ki, daha sonra bu playbook`u dəyişdirək. Ancaq bunu düzgün variable ilə set edildikdə buna ehtiyac qalmır. Bununlada həmişə playbook dəyişməsinin qarşısını alırıq. Variable bizim işimizi asanlaşdırır. Ansible variable çox yerdə istifadə olunur. Fact`lar da variable`nin bir növüdür. Linux`da variable üçün bashrc, profile faylına əlavə edirdik. Ansible`da isə bunu başqa yerlərə yazırıq.

Set-Variable

1)Ən az yazılan yer inventory`ə yazılır. Bu `best practise` sayılır. Təvsiyə olunmur.

```
vim /root/ansible/ingress
```

```
[prod]
192.168.149.136 app: murad

[test]
192.168.149.128
192.168.149.129

[webserver:children]
prod
```

2)Playbook daxilində də variable set edə bilərik.

```
vim var.yml
```

```
---
- name: 1st task
  hosts: proxy
  vars:
    - app: payment
    - env: prod
  tasks:
```

```
vim var.yml
```

```
---
- name: play for variables
  hosts: prod
  gather_facts: no
  vars:
    app: payment
    cluster: prod
  tasks:
    - name: debug for 1st variable
      debug:
        msg: 1st env is {{ app }}
    - name: debug for 2nd variable
      debug:
        msg: "{{ cluster }}"
```

vars-dictionary sayılır. - ilə yaza bilərik, dictionary olduğu üçün – sizdə yaza bilərik.

Birinci variable yazılırsa dırnaq içərisində yazılmalıdır.

3)Playbook daxilində fərqli cür də variable göstərə bilər. Daha `best practise` sayılır.Əgər playbook daxilində həm vars: həm də vars_files: verilərsə üstünlük vars_files: -dədir. İmtahan üçün ən ideali budur.

```
---
- name: play for variables
  hosts: all
  gather_facts: no

  vars_files:
    - vars/env.yml

  tasks:

    - name: debug for 1st variable
      debug:
        msg: 1st env is {{ app }}

    - name: debug for 2nd variable
      debug:
        msg: "{{ cluster }}"
```

Sonluğunun yml, txt olması fərq etmir.

mkdir vars/

vim vars/env.yml

```
app: price
cluster: test
~
~
~
~
~
```

vim var.yml

```
---
- name: play for variables
  hosts: prod
  gather_facts: no

  vars_files:
    - vars/env.yml
    - vars/env2.yml

  tasks:

    - name: debug for 1st variable
      debug:
        msg: 1st env is {{ app }}

    - name: debug for 2nd variable
      debug:
        msg: "{{ cluster }}"

    - name: debug for 3rd variable
      debug:
        msg: "{{ user }}"
```

İkinci vars_files yazıla bilər.

4)Hər qrup üçün ayrı variable təyin edək.

```
mkdir group_vars
vim group_vars/prod
```

```
app: tomcat
cluster: 25.25.36.36
~
```

vim 2varyml

```
---
- name: play1
  hosts: all
  gather_facts: no

  tasks:
    - name: debug1
      debug:
        msg: "{{app}}"
    - name: debug2
      debug:
        msg: cluster ip is {{ cluster }}
```

5)Hostlara görə də variable təyin edə bilərik.

```
mkdir host_vars
vim host_vars/192.168.149.136
```

```
app: ip
cluster: mysql
~
```

Group və host variable`ində üstünlük host-dadır.

shell>vars_files>vars>host_vars>group_vars>inventory

6)Shell-dən istifadə edərək variable təyin edək. Daha çox iş mühitində istifadə olunur.

```
ansible-playbook 3varyaml -e 'app=murad cluster=ali'
```

```
ok: [192.168.149.129] => {
  "msg": "app is murad"
}
ok: [192.168.149.136] => {
  "msg": "app is murad"
}

TASK [debug2] *****
ok: [192.168.149.128] => {
  "msg": "ali"
}
ok: [192.168.149.129] => {
  "msg": "ali"
}
ok: [192.168.149.136] => {
  "msg": "ali"
}
```

Example Variable Task

vim ders4.yml

```
--  
- name: play1  
  gather_facts: no  
  hosts: prod  
  vars:  
    pkg: httpd  
    role: http  
    pkg2: firewalld  
    user: reshad015  
    srv: httpd  
    hm: /home/reshad015
```

```
tasks:  
- name: paketleri install et  
  ansible.builtin.yum:  
    name:  
      - "{{pkg}}"  
      - "{{pkg2}}"  
    state: latest  
- name: servisi start et  
  ansible.builtin.service:  
    name: "{{srv}}"  
    state: started  
- name: firewalla elave et  
  ansible.posix.firewalld:  
    service: "{{role}}"  
    permanent: true  
    state: enabled  
    immediate: true
```

```
- name: useri yarat  
  ansible.builtin.user:  
    name: "{{user}}"  
    comment: test {{user}}  
    home: "{{hm}}"  
- debug:  
  msg: yaradilan user {{user}} , home folderi {{hm}}
```