Lesson 05 Demo 06

Working with Ansible Filters

Objective: To work with predefined and custom Ansible filters for manipulating data and transforming it into the desired format

Tools required: Linux terminal

Prerequisites: None

Steps to be followed:

1. Create the initial directory structure
2. Create a playbook to use predefined Ansible filters
3. Create a playbook to use custom Ansible filters
4. Run the playbooks

Step 1: Create the initial directory structure

1.1 Run the following command to create a folder:
mkdir ansible_filters_demo

```
poojahksimplile@ip-172-31-36-118:~$ mkdir ansible_filters_demo
```

1.2 Run the following command to navigate inside the **ansible_filters_demo** folder:
cd ansible_filters_demo

```
poojahksimplile@ip-172-31-36-118:~$ cd ansible_filters_demo
```

Step 2: Create a playbook to use predefined Ansible filters

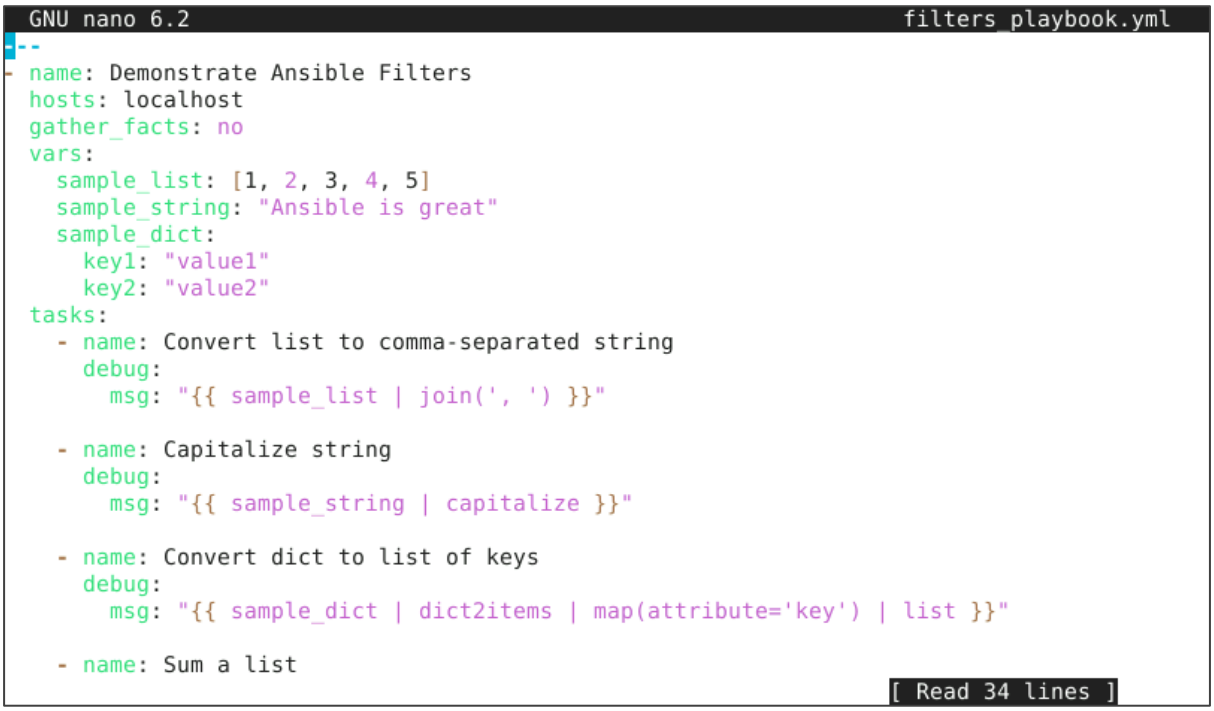
2.1 Run the following command to create a YAML file:
nano filters_playbook.yml

```
poojahksimplile@ip-172-31-36-118:~/ansible_filters_demo$ nano filters_playbook.yml
```

2.2 Enter the below script into the **filters_playbook.yml** file:

```
---
- name: Demonstrate Ansible Filters
  hosts: localhost
```

```
gather_facts: no
vars:
  sample_list: [1, 2, 3, 4, 5]
  sample_string: "Ansible is great"
  sample_dict:
    key1: "value1"
    key2: "value2"
tasks:
  - name: Convert list to comma-separated string
    debug:
      msg: "{{ sample_list | join(', ') }}"

  - name: Capitalize string
    debug:
      msg: "{{ sample_string | capitalize }}"

  - name: Convert dict to list of keys
    debug:
      msg: "{{ sample_dict | dict2items | map(attribute='key') | list }}"

  - name: Sum a list
    debug:
      msg: "{{ sample_list | sum }}"

  - name: Sort a list
    debug:
      msg: "{{ sample_list | sort(reverse=True) }}"

  - name: Check if item is in list
    debug:
      msg: "{{ 'yes' if 3 in sample_list else 'no' }}"
```



Step 3: Create a playbook to use custom Ansible filters

3.1 Run the following command to create a folder:
mkdir filter_plugins

```
poojahksimplile@ip-172-31-36-118:~/ansible_filters_demo$ mkdir filter_plugins
```

3.2 Run the following command to create a Python file for defining a custom Ansible filter:
nano filter_plugins/my_custom_filters.py

```
poojahksimplile@ip-172-31-36-118:~/ansible_filters_demo$ nano filter_plugins/my_custom_filters.py
```

3.3 Enter the below script into the **filter_plugins/my_custom_filters.py** file:

```
def multiply(value, by):
    return value * by

class FilterModule(object):
    def filters(self):
        return {
            'multiply': multiply
```

}

```
  GNU nano 6.2                                        filter_plugins/my_custom_filters.py
def multiply(value, by):
    return value * by

class FilterModule(object):
    def filters(self):
        return {
            'multiply': multiply
        }
```

3.4 Run the following command to create another YAML file:
nano custom_filters_playbook.yml

```
poojahksimplile@ip-172-31-36-118:~/ansible_filters_demo$ nano custom_filters_playbook.yml
```

3.5 Enter the below script into the **custom_filters_playbook.yml** file:

```
---
- name: Demonstrate Custom Filter Method
  hosts: localhost
  gather_facts: no
  vars:
    number: 5
    multiplier: 3
  tasks:
    - name: Multiply a number using custom filter
      debug:
        msg: "{{ number | multiply(multiplier) }}"
```

```
  GNU nano 6.2                                              custom_filters_playbook.yml
---
- name: Demonstrate Custom Filter Method
  hosts: localhost
  gather_facts: no
  vars:
    number: 5
    multiplier: 3
  tasks:
    - name: Multiply a number using custom filter
      debug:
        msg: "{{ number | multiply(multiplier) }}"
```

Step 4: Run the playbooks

4.1 Execute the following command to observe the working of built-in filters:
ansible-playbook filters_playbook.yml

```
poojahksimplile@ip-172-31-36-118:~/ansible_filters_demo$ ansible-playbook filters_playbook.yml


PLAY [Demonstrate Ansible Filters] ********************************************************************************

TASK [Convert list to comma-separated string] *********************************************************************
ok: [localhost] => {
    "msg": "1, 2, 3, 4, 5"
}

TASK [Capitalize string] ******************************************************************************************
ok: [localhost] => {
    "msg": "Ansible is great"
}

TASK [Convert dict to list of keys] *******************************************************************************
ok: [localhost] => {
    "msg": [
        "key1",
        "key2"
    ]
}

TASK [Sum a list] *************************************************************************************************
ok: [localhost] => {
    "msg": "15"
}

TASK [Sort a list] ************************************************************************************************
```

4.2 Execute the following command to observe the working of the custom filter:
ansible-playbook custom_filters_playbook.yml

```
poojahksimplile@ip-172-31-36-118:~/ansible_filters_demo$ ansible-playbook custom_filters_playbook.yml


PLAY [Demonstrate Custom Filter Method] ***************************************************************************

TASK [Multiply a number using custom filter] **********************************************************************
ok: [localhost] => {
    "msg": "15"
}

PLAY RECAP ********************************************************************************************************
localhost                  : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

poojahksimplile@ip-172-31-36-118:~/ansible_filters_demo$
```

By following these steps, you have successfully used predefined and custom Ansible filters to manipulate data and transform it into the desired format.