Tutorial 3 Création du premier playbook.

Table des matières

Objectifs

Dans ce TP ,nous allons créer notre premier playbook. Un playbook permet de regrouper des tâches à réaliser sur nos machines hôtes.

Ainsi nous allons :

- Créer un playbook basique en utilisant le module ping et debug utilisé dans le TP02.
- Le lancer.
- Analyser les résultats.
- Ajouter un tache de création de fichier avec file pour tester l'idempotence
-

Ressources

- Environnement
- Temps : 60 mn.

Pré-Requis

- avoir mis en place l'environnement dans cloud shell.
  TP01 - Mise en place du Lab
- avoir configurer Ansible pour travailler avec notre environnement dans cloud shell.
  TP02 - Configurer Ansible

Énoncé

Etape 1 création du projet TP03 dans myWork.

On peut configurer et spécifier des preferences dans le fichier ansible.cfg. Celui ci peut se retrouver à plusieurs endroits,mais il est fortement conseillé de le faire dans le repertoire du projet.

1. copiez le projet TP02 en TP03. depuis la console d'ubuntu-c

```
cd /home/ansible/ansibleMelodie/myWork
cp -R ./TP02 TP03
```

ou copie dans solution correction

1. placez vous de le dossier nouvellement créé et tester les connexions.
    - faire du ping/pong avec les hôtes inscrits.

      ansible -i 00_inventory.yml all -m ping

2. copiez le fichier playbook 10_playbook.yml du projet template dans notre projet via la commande dans le terminal.

```
cp ../../templates/ansible_simple/10_playbook.yml 10_premierPlaybook.yml
```

Etape 2 Testons nos connexions dans un playbook avec le module PING.

1. création du jeu
    - dans le fichier 10_premierPlaybook remplacer :

```
---
# Les documents YAML commencent par le séparateur de document ---
# Le moins (-) dans YAML indique un élément de liste.
# Le playbook contient une liste de "jeu".
# Chaque jeu étant un dictionnaire.
- name: le nom de mon jeu
# Hosts: les systèmes cibles où notre jeu s'exécutera et les options avec
lesquelles il s'exécutera
  hosts:
# Vars: Les variables qui s'appliqueront à ce jeu, sur tous les systèmes
cibles
```

par

```
---
- name: tests de notre configuration.
  hosts: all
```

içi nous indiquons que notre jeu s'appelle "tests de notre configuration" et qu'il doit être joué sur toutes nos machines (control et managed node).
👁remarquer que nous n'avons pas indiqué de variables puisque nous le faisons avec les dossiers d'inventaire (group_vars et hosts_vars).

2. création de la tache pour tester la connexion.
   - dans le fichier 10_premierPlaybook remplacer :

```
 # Tasks: la liste des tâches qui seront exécutées dans ce jeu.
 tasks:
   - name: nom de la tâche
```

par

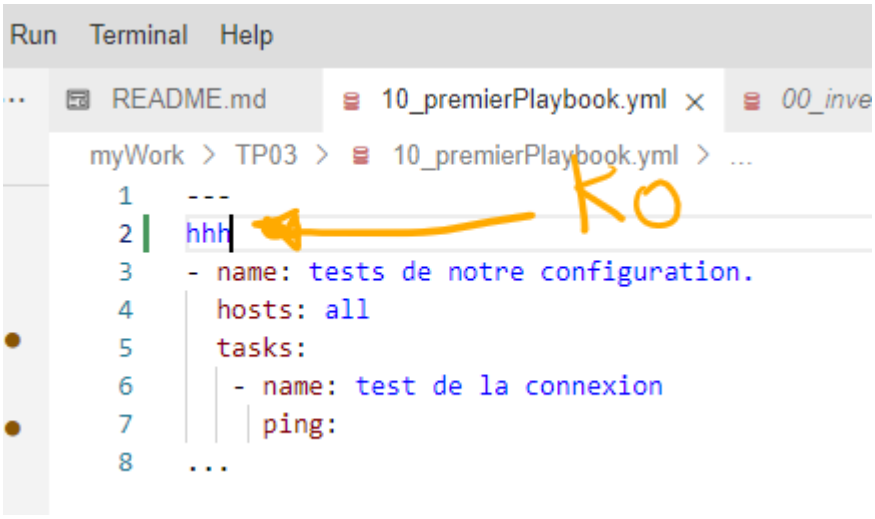
```
 tasks:
   - name: test de la connexion
     ping:
```

⚠ faites très attention à l'indentation Ansible et le format yaml sont très sensible.
👁Préférez les espaces pour indenter (deux espaces) à la touche de tabulation. ou copie dans solution correction

1. lancer votre playbook.
   `ansible-playbook 10_premierPlaybook.yml -i 00_inventory.yml`

```
ansible@ubuntu-c:~/ansibleMelodie/myWork/TP03$ ansible-playbook 10_premierPlaybook.yml -i 00_inventory.yml

PLAY [tests de notre configuration.] ***************************************************************************************

TASK [Gathering Facts] *****************************************************************************************************
ok: [ubuntu1]
ok: [ubuntu-c]
ok: [centos3]
ok: [centos1]
ok: [centos2]
ok: [ubuntu2]
ok: [ubuntu3]

TASK [test de la connexion] ************************************************************************************************
ok: [ubuntu1]
ok: [centos3]
ok: [centos2]
ok: [centos1]
ok: [ubuntu-c]
ok: [ubuntu2]
ok: [ubuntu3]

PLAY RECAP *****************************************************************************************************************
centos1                    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos2                    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos3                    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu-c                   : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu1                    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu2                    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu3                    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ansible@ubuntu-c:~/ansibleMelodie/myWork/TP03$
```

2. verifier la syntaxe de votre playbook. `ansible-playbook 10_premierPlaybook.yml -i 00_inventory.yml --syntax-check`
    - ajouter une erreur de syntaxe et relancer la commande




```
ansible@ubuntu-c:~/ansibleMelodie/myWork/TP03$ ansible-playbook 10_premierPlaybook.yml -i 00_inventory.yml --syntax-check
ERROR! We were unable to read either as JSON nor YAML, these are the errors we got from each:
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/ansible/ansibleMelodie/myWork/TP03/10_premierPlaybook.yml': line 3, column 7, but may
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

hhh
- name: tests de notre configuration.
      ^ here
ansible@ubuntu-c:~/ansibleMelodie/myWork/TP03$
```

Visual Studio Code

Etape 3 Testons nos variables dans un playbook avec le module DEBUG.

En vous aidant de l'étape 2 et du TP02 ajouter une nouvelle tâche qui permettra d'afficher le message **le contenu de ma variable est {{ maVariable }}**.
Affiche des contenus de variables ou messages

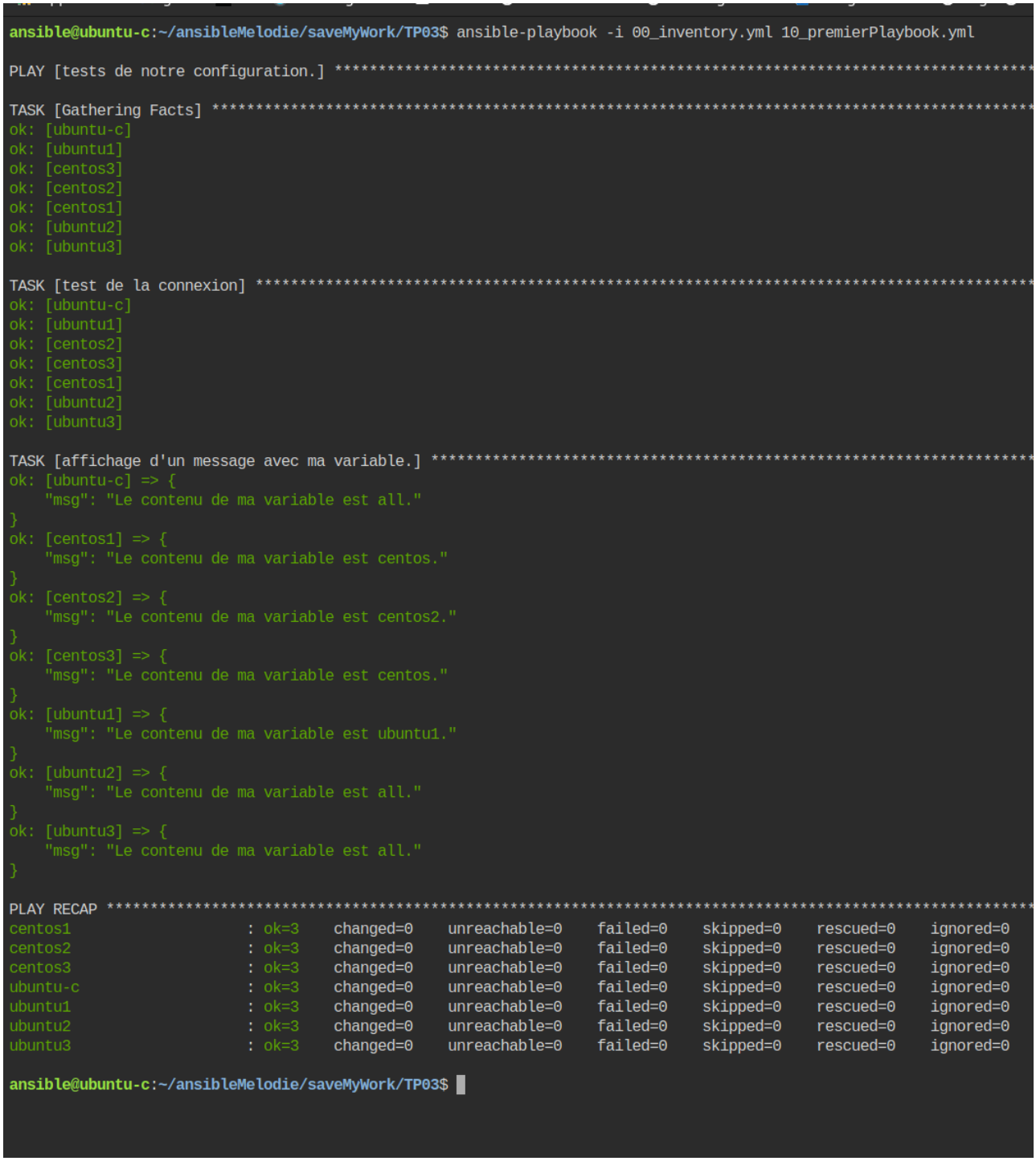
1. A la fin du fichier 10_premierPlaybook.yml ajouter la nouvelle tâche utilsan tle module DEBUG.

```
---
- name: tests de notre configuration.
  hosts: all
  tasks:
   - name: test de la connexion
     ping:
...
```

devient

```
---
- name: tests de notre configuration.
  hosts: all
  tasks:
   - name: test de la connexion
     ping:
   - name: affichage d'un message avec ma variable.
     debug:
      msg: Le contenu de ma variable est {{ maVariable }}.
...
```

1. lancer votre playbook.
   `ansible-playbook 10_premierPlaybook.yml -i 00_inventory.yml`



Etape 4 Testons l'idempotence en créant un dossier avec le module FILE.

Pour ce faire Nous allons ajouter un nouveau jeu (play) que nous ne lancerons que sur les machines hôtes (group **managed**).
Nous utiliserons le module FILE pour créer un répertoire.
Nous relancerons plusieurs fois pour se familiariser avec les concepts d'idempotence et statut de taches.

1. ajout d'un nouveau jeu restreint au groupe **managed** de l'inventaire.

- modifier le fichier 10_premierPlaybook.

```
---
- name: tests de notre configuration.
```

```
  hosts: all
  tasks:
   - name: test de la connexion
     ping:
   - name: affichage d'un message avec ma variable.
     debug:
      msg: Le contenu de ma variable est {{ maVariable }}.
...
```

devient

```
---
- name: tests de notre configuration.
  hosts: all
  tasks:
   - name: test de la connexion
     ping:
   - name: affichage d'un message avec ma variable.
     debug:
      msg: Le contenu de ma variable est {{ maVariable }}.

- name: creation d'un dossier ~/tmp.
  hosts: managed
  tasks:
...
```

- testons en lançant l'execution du playbook. `ansible-playbook 10_premierPlaybook.yml -i 00_inventory.yml`

```
}
ok: [ubuntu3] => {
    "msg": "Le contenu de ma variable est all."
}

PLAY [creation d'un dossier ~/tmp.] ************************************************************

TASK [Gathering Facts] *************************************************************************
ok: [ubuntu2]
ok: [ubuntu1]
ok: [centos1]
ok: [centos2]
ok: [centos3]
ok: [ubuntu3]

PLAY RECAP *************************************************************************************
centos1                    : ok=4    changed=0    unreachable=0    failed=0    skipped=0    r
centos2                    : ok=4    changed=0    unreachable=0    failed=0    skipped=0    r
centos3                    : ok=4    changed=0    unreachable=0    failed=0    skipped=0    r
```

Le second jeu est limité aux machines hôtes.

1. ajout de la tache pour créer le nouveau dossier temporaire `tmp` dans le dossier home (`~`).

```
---
- name: tests de notre configuration.
```

```
  hosts: all
  tasks:
   - name: test de la connexion
     ping:
   - name: affichage d'un message avec ma variable.
     debug:
      msg: Le contenu de ma variable est {{ maVariable }}.

- name: creation d'un dossier ~/tmp.
  hosts: managed
  tasks:
...
```

devient

```
---
- name: tests de notre configuration.
  hosts: all
  tasks:
   - name: test de la connexion
     ping:
   - name: affichage d'un message avec ma variable.
     debug:
      msg: Le contenu de ma variable est {{ maVariable }}.

- name: creation d'un dossier ~/tmp.
  hosts: managed
  tasks:
    - name: ajout du dossier.
      file:
        state: directory
        path: ~/tmp
...
```

- lancer l'execution du playbook. `ansible-playbook 10_premierPlaybook.yml -i 00_inventory.yml`

1. premier lancement.

```
TASK [ajout du dossier.] ********************************************************************************************
changed: [ubuntu2]
changed: [ubuntu1]
changed: [centos1]
changed: [centos2]
changed: [centos3]
changed: [ubuntu3]

PLAY RECAP **********************************************************************************************************
centos1                    : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos2                    : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos3                    : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu-c                   : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu1                    : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu2                    : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu3                    : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ansible@ubuntu-c:~/ansibleMelodie/saveMyWork/TP03$
```

2. second lancement.

```
TASK [ajout du dossier.] ***********************************************************************************************
ok: [ubuntu1]
ok: [ubuntu2]
ok: [centos3]
ok: [centos1]
ok: [centos2]
ok: [ubuntu3]

PLAY RECAP *************************************************************************************************************
centos1                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos2                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos3                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu-c                   : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu1                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu2                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu3                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ansible@ubuntu-c:~/ansibleMelodie/saveMyWork/TP03$
```

3. suppression sur `centos1` du dossier.

```
ansible -i 00_inventory.yml centos1 -m ansible.builtin.file -a "path=~/tmp state=absent"
```

```
[WARNING]: Could not match supplied host pattern, ignoring: centos1
ansible@ubuntu-c:~/ansibleMelodie/saveMyWork/TP03$ ansible -i 00_inventory.yml centos1 -m ansible.builtin.file -a "path=~/tmp state=absent"
centos1 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": true,
    "path": "/home/ansible/tmp",
    "state": "absent"
}
ansible@ubuntu-c:~/ansibleMelodie/saveMyWork/TP03$ ^C
ansible@ubuntu-c:~/ansibleMelodie/saveMyWork/TP03$
```

1. nouveau lancement

```
TASK [ajout du dossier.] ***********************************************************************************************
ok: [ubuntu1]
ok: [ubuntu2]
ok: [centos2]
ok: [centos3]
changed: [centos1]
ok: [ubuntu3]

PLAY RECAP *************************************************************************************************************
centos1                    : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos2                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
centos3                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu-c                   : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu1                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu2                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu3                    : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ansible@ubuntu-c:~/ansibleMelodie/saveMyWork/TP03$
```

Conclusion et feed-back

Correction

💡💡💡💡Idées

-