

Mini-projet Final

Repository

<https://github.com/infradev4/roleWordpress.git>

<https://github.com/infradev4/roleDocker.git>

Créez un rôle permettant de déployer Wordpress

- Vous avez reçu la demande d'une autre équipe qui souhaiterait utiliser un de vos playbook de déploiement de WordPress, mais sous forme de rôle car sous cette forme ils pourront mieux variabiliser et adapter à leur situation
- Leur objectif est que votre rôle possède un playbook tests afin de leur permettre de tester rapidement votre rôle et ainsi l'intégrer à leur process de déploiement, exactement comme le rôle wordpress
- Utilisez le rôle docker précédemment crée afin de créer un nouveau rôle qui permettra de déployer l'application wordpress à l'aide de deux containers docker (mysql + wordpress)
- Déployez vos différents container à l'aide du module docker_container, ces containers devront appartenir à un meme réseau « wordpress » que vous devez au préalable créer à partir du module ansible « docker-network »
- Variabilisez un maximum de paramètres de ce role afin qu'il puisse etre utilisé par différentes entreprises fonction du besoin, on devra pouvoir personnaliser lors du déploiement :
 - Le nom du réseau dans lequel sera crée les container
 - Le nom des container
 - Le port sur lequel consommer l'application
 - Le nom du volume dans lequel sera sauvegardé le BDD mysql (mettre en place la persistance)
- A la fin de votre travail, poussez votre rôle sur github et sur la galaxy ansible (wordpress_role) et envoyez nous votre rapport détaillé de mise en œuvre en PDF via l'intranet (Partage de documents)

Structure de mon projet

/home/ubuntu/roles

```
docker/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

wordpress/
├── defaults
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── tests
│   ├── inventory
│   └── test.yml
```

Variabilisation de mon role wordpress

```
wordpress/
├── defaults
│   └── main.yml
```

`vi wordpress/defaults/main.yml`

```
# Docker Create a network
network_one: wordpress

# Docker wordpress settings
wordpress_container_name: wordpress
wordpress_exposed_port: 80
wordpress_container_data: wordpressData

# Docker mysql settings
mariadb_container_name: mysql
```

```
mysql_exposed_port: 3306
mysql_root_password: root123!
mysql_database: db_wordpress
mysql_user: wordpress
mysql_password: wordpress123!
mysql_container_data: mysqlData
```

Création du fichier de tâches (tasks)

```
wordpress/
├── tasks
│   └── main.yml
```

`docker_network:`
 Créez/supprimez des réseaux Docker et connectez-y des conteneurs.

<https://docs.docker.com/network/links/>
https://docs.ansible.com/ansible/2.9/modules/docker_container_module.html

```
links:
  - "{{ mariadb_container_name }}:mysql_host"
```

Les liens permettent aux conteneurs de se découvrir et de transférer en toute sécurité des informations sur un conteneur à un autre conteneur. Lorsque vous configurez un lien, vous créez un conduit entre un conteneur source et un conteneur destinataire. Le destinataire peut alors accéder à des données sélectionnées sur la source. Pour créer un lien, vous utilisez l'indicateur `--link`. Tout d'abord, créez un nouveau conteneur, cette fois contenant une base de données.

`vi wordpress/tasks/main.yml`

```
---
# tasks file for wordpress_role
- name: Create a network
  docker_network:
    name: "{{ network_one }}"

- name: start mariadb containers
  docker_container:
    name: "{{ mariadb_container_name }}"
```

```

    image: "{{ mariadb_container_name }}"
    state: started
    restart_policy: always
    env:
      MYSQL_ROOT_PASSWORD: "{{ mysql_root_password }}"
      MYSQL_DATABASE: "{{ mysql_database }}"
      MYSQL_USER: "{{ mysql_user }}"
      MYSQL_PASSWORD: "{{ mysql_password }}"
    ports:
      - "{{ mysql_exposed_port }}:3306"
    volumes:
      - "{{ mysql_container_data }}:/var/lib/mysql"

- name: start wordpress containers
  docker_container:
    name: "{{ wordpress_container_name }}"
    image: "{{ wordpress_container_name }}"
    state: started
    restart_policy: always
    env:
      WORDPRESS_DB_HOST: mysql_host
      WORDPRESS_DB_USER: "{{ mysql_user }}"
      WORDPRESS_DB_PASSWORD: "{{ mysql_password }}"
      WORDPRESS_DB_NAME: "{{ mysql_database }}"
    ports:
      - "{{ wordpress_exposed_port }}:80"
    links:
      - "{{ mariadb_container_name }}:mysql_host"
    volumes:
      - "{{ wordpress_container_data }}:/var/www/html"

- name: Add a container to a network, leaving existing containers connected
  docker_network:
    name: "{{ network_one }}"
    connected:
      - "{{ mariadb_container_name }}"
      - "{{ wordpress_container_name }}"
    appends: yes

```

Playbook tests

```

wordpress/
├── tests
│   ├── inventory
│   └── test.yml

```

```
vi wordpress/tests/inventory
```

```
---  
localhost
```

vi wordpress/tests/test.yml

```
---  
- hosts: localhost  
  remote_user: root  
  become: true  
  vars:  
    ansible_connection: local  
  roles:  
    - wordpress
```

ansible-playbook pour le role Docker & Wordpress

```
roles/  
├─ docker  
├─ wordpress  
├─ hosts.yml  
└─ docker.yml
```

vi docker.yml

```
---  
- name:  
  hosts: prod  
  become: true  
  roles:  
    - docker  
    - wordpress
```

vi hosts.yml

```
---  
all:  
  children:  
    ansible:  
      hosts:  
        localhost:  
          ansible_connection: local  
          ansible_user: ubuntu  
          ansible_password: ubuntu
```

```
    hostname: AnsibleMaster
prod:
  hosts:
    worker01:
      ansible_host: 172.31.14.210
      ansible_user: ubuntu
      ansible_password: ubuntu
      ansible_ssh_common_args: '-o StrictHostKeyChecking=no'
      hostname: AnsibleWorker01

    worker02:
      ansible_host: 172.31.0.185
      ansible_user: ubuntu
      ansible_password: ubuntu
      ansible_ssh_common_args: '-o StrictHostKeyChecking=no'
      hostname: AnsibleWorker02
```

Création dy fichier galaxy

```
roles/
├─ docker
├─ wordpress
├─ hosts.yaml
├─ docker.yaml
└─ galaxy.yaml
```

vi galaxy.yaml

```
---
namespace: community
name: wordpress
version: 1.0.0
readme: README.md
authors:
  - Oussama ZAID
description: Install wordpress with docker
license_file: COPYING
tags:
  - v1
repository: https://github.com/infradev4/roleWordpress.git
documentation: https://github.com/infradev4/roleWordpress.git
homepage: https://github.com/infradev4/roleWordpress.git
issues: https://github.com/infradev4/roleWordpress.git
build_ignore:
  - .gitignore
```

Création dy fichier main.yml de meta

```
wordpress/
├── meta
└── main.yml
```

```
vi main.yml
```

```
---
galaxy_info:
  author: Oussama ZAID
  description: Setup simple wordpress site using docker container
  license: COPYING
  min_ansible_version: core 2.12.1
```

```
ubuntu@AnsibleMaster:~/roles$ ansible-playbook -i hosts.yaml docker.yaml
```

```
PLAY [prod]
*****
*****

TASK [Gathering Facts]
*****
*****

ok: [worker02]
ok: [worker01]

TASK [docker : push script file]
*****
*****

ok: [worker02]
ok: [worker01]

TASK [docker : download pip script]
*****
*****

ok: [worker01]
ok: [worker02]

TASK [docker : install python-pip]
*****
*****

skipping: [worker01]
skipping: [worker02]

TASK [docker : Install docker python]
*****
```

```

*****
skipping: [worker01]
skipping: [worker02]

TASK [docker : run the script]
*****
*****

skipping: [worker01]
skipping: [worker02]

TASK [wordpress : Create a network]
*****
*****

ok: [worker02]
ok: [worker01]

TASK [wordpress : start mariadb containers]
*****
*****

ok: [worker02]
ok: [worker01]

TASK [wordpress : start wordpress containers]
*****
*****

changed: [worker02]
changed: [worker01]

TASK [wordpress : Add a container to a network, leaving existing containers
connected] *****

changed: [worker02]
changed: [worker01]

PLAY RECAP
*****
*****

worker01          : ok=7    changed=2    unreachable=0    failed=0
skipped=3         rescued=0    ignored=0
worker02          : ok=7    changed=2    unreachable=0    failed=0
skipped=3         rescued=0    ignored=0

```

Test Worker01

```

ubuntu@AnsibleWorker01:~$ docker ps
IMAGE          PORTS
wordpress     0.0.0.0:80->80/tcp
mysql         0.0.0.0:3306->3306/tcp, 33060/tcp
NAMES
wordpress
mysql

```

Adressage Worker01


```
ubuntu@AnsibleWorker01:~$ docker inspect wordpress | grep -i "IPAddress"
    "SecondaryIPAddresses": null,
    "IPAddress": "172.17.0.3",
        "IPAddress": "172.17.0.3",
        "IPAddress": "172.18.0.3",
```

Adressage Worker02

```
ubuntu@AnsibleWorker01:~$ docker inspect mysql | grep -i "IPAddress"
    "SecondaryIPAddresses": null,
    "IPAddress": "172.17.0.2",
        "IPAddress": "172.17.0.2",
        "IPAddress": "172.18.0.2",
```

volume Worker01 wordpress

```
ubuntu@AnsibleWorker01:~$ docker inspect wordpress | grep -i "Source"
    "Source": "/var/lib/docker/volumes/wordpressData/_data",
```

volume Worker01 mysql

```
ubuntu@AnsibleWorker01:~$ docker inspect mysql | grep -i "Source"
    "Source": "/var/lib/docker/volumes/mysqlData/_data",
```

Afficher les liens (links) conteneurs en cours

```
ubuntu@AnsibleWorker01:~$ docker inspect wordpress -f "{{ .HostConfig.Links }}"
[/mysql:/wordpress/mysql_host]
```

Répertorier toutes les liaisons de port

```
ubuntu@AnsibleWorker01:~$ docker inspect --format='{{range $p, $conf :=
.NetworkSettings.Ports}} {{$p}} -> {{(index $conf 0).HostPort}} {{end}}' wordpress

80/tcp -> 80
```

Génère les entrées Name et Driver séparées par deux points (😊) pour tous les volumes :

```
ubuntu@AnsibleWorker01:~$ docker volume ls --format "{{.Name}}: {{.Driver}}"  
mysqlData: local  
wordpressData: local
```



English (United States)

Afrikaans

العربية

العربية المغربية

অসমীয়া

Azərbaycan dili

گۆنئی آذربایجان

Беларуская мова

Български

বাংলা

བོད་སྐད་

Bosanski

Català

Cebuano

Čeština

Cymraeg

Dansk

Deutsch

Deutsch (Schweiz, Du)

Deutsch (Schweiz)

Deutsch (Österreich)

Continue

roleWordpress

Public

Star

Updated 8 minutes ago

roleDocker

Public

Star

Shell Updated 11 minutes ago

The screenshot shows the Ansible Galaxy 'My Content' page for the user 'infradev4'. The page displays a list of repositories under the 'infradev4' namespace. Two repositories are visible: 'docker_role' (Ansible docker role) with a 4.8/5 score and 'rolewordpress' (Setup simple wordpress site using docker container) with a 5/5 score. Both show a success status. The interface includes a sidebar with navigation options like Home, Search, Community, My Content, and My Imports. The top header shows the user's profile and navigation links like About, Help, and Documentation.

galaxy

Docker 4.8/5 Score

https://galaxy.ansible.com/infradev4/docker_role

ansible-galaxy install infradev4.docker_role

Wordpress 5/5 Score

<https://galaxy.ansible.com/infradev4/rolewordpress>

ansible-galaxy install infradev4.rolewordpress