06

Mãos na massa: Facilitando o reuso

Chegou a hora de você executar o que foi visto na aula! Para isso, execute os passos listados abaixo.

1) Agora que você já tem o Ansible configurado, você precisa pensar em como melhorar o código para aumentar o seu reuso. Reorganize-o em **roles**, para isso crie um diretório **roles** com 3 subdiretórios: **mysql**, **webserver** e **wordpress**.

2) A primeira *role* vai lidar com o **MySQL**, crie o arquivo no seguinte caminho: **roles/mysql/tasks/main.yml**. Nele, coloque o trecho de código específico dessa *role*:

```
---
- name: 'Instala pacotes de dependencia do sistema operacional'
  apt:
    name: "{{ item }}"
    state: latest
  become: yes
  with_items:
    - mysql-server-5.6
    - python-mysqldb

- name: 'Cria o banco do MySQL'
  mysql_db:
    name: "{{ wp_db_name }}"
    login_user: root
    state: present

- name: 'Cria o usuário do MySQL'
  mysql_user:
    login_user: root
    name: "{{ wp_username }}"
    password: "{{ wp_user_password }}"
    priv: "{{ wp_db_name }}.*:ALL"
    state: present
    host: "{{ item }}"
  with_items:
   - 'localhost'
   - '127.0.0.1'
   - "{{ wp_host_ip }}"

- name: 'Configura MySQL para aceitar conexões remotas'
  copy:
    src: 'files/my.cnf'
    dest: '/etc/mysql/my.cnf'
  become: yes
  notify:
    - restart mysql
```

3) O *handler* dessa *role* ficará em **roles/mysql/handlers/main.yml**, com o seguinte conteúdo:

```
---
- name: restart mysql
```

```
    service:
      name: mysql
      state: restarted
    become: yes
```

4) Feito isso, em **provisioning.yml**, apague esses trechos que foram externalizados nos dois arquivos anteriores e invoque a *role* criada para o *database*:

```
---
- hosts: database
  roles:
    - mysql

# Outros HOSTS omitidos
```

5) Depois da primeira *role* implementada, faça de maneira análoga para **webserver**.

Crie o arquivo **roles/webserver/tasks/main.yml** com o conteúdo a seguir. Perceba que esse e todos o conteúdos desse exercício já haviam sido definidos em **provisioning.yml**, basta copiá-lo e removê-lo do arquivo de origem:

```
---
- name: 'Instala pacotes de dependencia do sistema operacional'
  apt:
    name: "{{ item }}"
    state: latest
  become: yes
  with_items:
    - php5
    - apache2
    - libapache2-mod-php5
    - php5-gd
    - libssh2-php
    - php5-mcrypt
    - php5-mysql
```

Feito isso, inclua essa *role* dentro de **provisioning.yml**:

```
# código anterior omitido...
- hosts: wordpress
  handlers:
    # código do handler omitido...
  roles:
    - webserver
```

6) Agora, cuide da instalação do WordPress em uma nova *role*. Dentro de **roles/wordpress/tasks/main.yml**, você terá o seguinte conteúdo:

```
---
- name: 'Baixa o arquivo de instalacao do Wordpress'
  get_url:
    url: 'https://wordpress.org/latest.tar.gz'
```

```
    dest: '/tmp/wordpress.tar.gz'

- name: 'Descompacta o wordpress'
  unarchive:
    src: '/tmp/wordpress.tar.gz'
    dest: /var/www/
    remote_src: yes
  become: yes

- copy:
    src: "{{ wp_installation_dir }}/wp-config-sample.php"
    dest: "{{ wp_installation_dir }}/wp-config.php"
    remote_src: yes
  become: yes

- name: 'Configura o wp-config com as entradas do banco de dados'
  replace:
    path: "{{ wp_installation_dir}}/wp-config.php"
    regexp: "{{ item.regex }}"
    replace: "{{ item.value }}"
  with_items:
    - { regex: 'database_name_here', value: "{{ wp_db_name }}"}
    - { regex: 'username_here', value: "{{ wp_username }}"}
    - { regex: 'password_here', value: "{{ wp_user_password }}"}
    - { regex: 'localhost', value: "{{ wp_db_ip }}"}
  become: yes

- name: 'Configura Apache para servir Wordpress'
  template:
    src: 'templates/000-default.conf.j2'
    dest: '/etc/apache2/sites-available/000-default.conf'
  become: yes
  notify:
    - restart apache
```

7) Você precisará do *handler* para essa *role*. Crie o arquivo **roles/wordpress/handlers/main.yml** com o seguinte conteúdo:

```
---
- name: restart apache
  service:
    name: apache2
    state: restarted
  become: yes
```

8) Agora, **proviosning.yml** ficará como a seguir:

```
---
- hosts: database
  roles:
    - mysql

- hosts: wordpress
  roles:
```

```
    - webserver
    - wordpress
```

9) Para melhorar ainda mais, coloque **templates** e **files** dentro das *roles*. Embora esteja funcionando sem esse cuidado, essa prática de deixar esses arquivos no nível do projeto não facilitará o reuso em projetos futuros.

Então, mova o arquivo **files/my.cnf** para o diretório (que deve ser criado) **roles/mysql/files**.

10) Mova também o template **templates/000-default.conf.j2** para o diretório (que também deve ser criado) **roles/wordpress/templates**

11) Delete os diretórios que ficaram vazios: **files** e **templates**

12) Para deixar ainda mais expressivo e evitar problemas da dependência do WordPress com a *role* do *webserver* crie o arquivo **roles/wordpress/meta/main.yml** com o seguinte conteúdo:

```
---
dependencies:
  - webserver
```

E remova essa role do **provisoning.yml**, que terá como conteúdo final:

```
---
- hosts: database
  roles:
    - mysql

- hosts: wordpress
  roles:
    - wordpress
```

13) Para melhorar ainda mais, defina valores padrões para as *roles*. Para isso, externalize os IPs liberados para acesso ao MySQL em um arquivo **main.yml**, que ficará em **roles/mysql/defaults**, e terá o seguinte conteúdo:

```
---
wp_host_ip:
  - localhost
  - '127.0.0.1'
```

Lembre-se de **remover** esses dois IPs de **roles/mysql/tasks/main.yml**:

```
- name: 'Cria o usuário do MySQL'
  mysql_user:
    login_user: root
    name: "{{ wp_username }}"
    password: "{{ wp_user_password }}"
    priv: "{{ wp_db_name }}.*:ALL"
    state: present
    host: "{{ item }}"
```

```
    with_items:
     - "{{ wp_host_ip }}"
```

14) Por fim, rode o *Playbook*:

```
ansible-playbook -i hosts provisioning.yml
```