- 角色（roles）
  - 1. role的基本构成
  - 2. 在playbook中使用roles
  - 3. `pre_tasks` 和 `post_tasks`
  - 4. role的依赖
- Ansible Galaxy

角色（roles）

在Ansible中，role是将playbook分割为多个文件的主要机制。它大大简化了复杂playbook的编写，同时还使得它们非常易于复用。

1. role的基本构成

roles文件组织结构示例：

```
group_vas/
site.yml
webservers.yml
roles/
   common/
     files/
     templates/
     tasks/
     handlers/
     vars/
     defaults/
     meta/
   webservers/
     files/
     templates/
     tasks/
     handlers/
     vars/
     defaults/
     meta/
```

roles各目录的作用及可用的文件：

- files：用于存放一些非模板文件的文件，如https证书等。
- tempaltes：用于存放角色相关的Jinja2模板文件，当使用角色相关的模板时，如未明确指定模板路径，则默认使用此目录中的模板
- tasks：角色所要执行的所有任务文件都存放于此，包含一个主文件main.yml，可以在主文件中通过include的方式引入其他任务文件

- handlers：用于定义角色中需要调用 的handlers，包含一个主配置文件main.yml，可通过include引入其他的handlers文件。
- vars：用于定义此角色用到的变量，包含一个主文件main.yml
- meta：用于存储角色的元数据信息，这些元数据用于描述角色的相关属性，包括作者，角色的主要作用，角色的依赖关系等。默认这些信息会写入到当前目录下的main.yml文件中
- defaults：除了vars目录，defaults目录也用于定义此角色用到的变量，与vars不同的是，defaults中定义的变量的优先级最低。

创建role的步骤如下：

1. 创建以roles命名的目录
2. 在roles目录中分别创建角色名称命名的目录，如websrvs等
3. 在每个角色命名的目录中分别创建files、handlers、meta、tasks、teamplates和vars目录，用不到的目录可以创建为空目录，也可以不创建。
4. 在playbook文件中，调用各角色

> 需要说明的是，以上目录并不都是必须的，如果你的roles当中并不需要用到某一个目录，也可以不用创建，比如我们将所有的变量都放到defaults中，则可以不需要vars目录，如果未用到模板文件，则不需要templates目录。

2. 在playbook中使用roles

基本引用的方法：

```
- hosts: webservers
  roles:
     - common
     - webserver
```

也可以通过如下方法引用时带入变量：

```
- hosts: webservers
  roles:
    - common
    - { role: foo_app_instance, dir: '/opt/a',  port: 5000 }
    - { role: foo_app_instance, dir: '/opt/b',  port: 5001 }
```

还可以在引用时使用条件语句：

```
- hosts: webservers
  roles:
    - { role: some_role, when: "ansible_os_family == 'RedHat'" }
```

下面也是一个带入变量的示例：

```
- hosts: webservers
  roles:
    - role: database
      vars:
        database_name: {{ db_name }}
        database_user: {{ db_pass }}
    - role: webserver
      vars:
        live_hostname: web1
        domains:
          - example.com
          - www.example.com
```

> 注1： 在引用roles时，在roles下使用vars传入的变量是全局的，即多个role都会共享这一变量

> 注2： 在roles当中，通过命令行传入的变量优先级最高，其次是定义在vars目录下的变量，再次是使用vars定义在引入的roles文件中的变量

3. `pre_tasks` 和 `post_tasks`

如果在执行一个role时，需要在其前或其后依然要执行某些任务，我们可以使用 `pre_tasks` 及 `post_tasks` 来声明。 `pre_tasks` 是在 `role` 之前执行，而 `post_tasks` 则在role之后执行：

```
- name: deply webservers
  host: webservers
  vars_files:
    - secrets.yml
  pre_tasks:
    - name: update yum cache
      yum: update_cache=yes
  roles:
    - role: apache
      database_host: {{ hostvars.db.ansible_eth0.ipv4.address }}
      domains:
        - exampel.com
        - www.example.com
  post_tasks:
    - name: print something
      shell: echo "The roles have been updated!"
```

4. role的依赖

如果当前role在执行前需要依赖另一个role，我们可以在roles的meta目录中的main.yml中定义role的依赖关系。

示例1：

```
# roles/webservers/meta/main.yml
dependencies:
  - { role: common, some_parameter: 3 }
  - { role: apache, port: 80 }
  - { role: postgres, dbname: blarg, other_parameter: 12 }
```

示例2：

```
dependencies:
  - {role: ntp, ntp_server=ntp.ubuntu.com}
  - {role: web}
  - {role: memcached}
```

Ansible Galaxy

ansible-galaxy是一个工具，我们可以利用它快速的创建一个标准的roles目录结构，还可以通过它在https://galaxy.ansible.com上下载别人写好的roles，直接拿来用。

通过ansible-galaxy初始化一个roles的目录结构，方法如下：

```
ansible-galaxy init my_new_role
```

安装别人写好的roles：

```
ansible-galaxy install -p /etc/ansible/roles bennojoy.mysql
```

列出已安装的roles：

```
ansible-galaxy list
```

查看已安装的roles信息：

```
ansible-galaxy info bennojoy.mysql
```

卸载roles：

```
ansible-galaxy remove bennojoy.mysql
```