05.Ansible Task控制

0. overview

1.Playbook条件语句

when语句传送门

判断在Ansible任务中的使用频率非常高。比如yum模块可以检测软件包是否已被安装，而在这个过程中我们不用做太多的人工干预。
但是也有部分任务需要进行判断，比如: web服务器角色都需要安装nginx仓库，但其他的服务器角色并不需要，此时就会用到when判断。
比如: Centos与Ubuntu系统都需要安装httpd服务，那么就需要使用when判断主机系统，然后调用不同的模块执行。

实践案例一、根据不同操作系统，安装相同的软件包

```
[root@m01 /server/playbook]# cat 08_system.yml
- hosts: centubt
  tasks:
    - name: CentOS install cowsay
      yum:
        name:
```

```
        - cowsay
        state: present
      when: ( ansible_distribution == "CentOS" )   #只有
centos系统才安装 cowsay

    - name: Ubuntu install cmatrix
      apt:
        name:
        - cmatrix
        state: present
      when: ( ansible_distribution == "Ubuntu" )    #只有
ubuntu系统 才安装 cmatrix
[root@m01 /server/playbook]#

# 在when条件中
# 加上 小括号
# 在when中变量直接使用 不需要加上 {{}}
```

2.执行playbook

```
[root@m01 /server/playbook]# ansible-playbook -i hosts
08_system.yml

PLAY [centubt]
**********************************************************
**********************************************************

TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.9]
ok: [172.16.1.42]

TASK [CentOS install cowsay]
**********************************************************
********************************************
skipping: [172.16.1.42]
ok: [172.16.1.9]
```

```
TASK [Ubuntu install cmatrix]
*****************************************************************************************************
skipping: [172.16.1.9]
changed: [172.16.1.42]

PLAY RECAP
*********************************************************************************************************************************
172.16.1.42                : ok=2    changed=1
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.9                 : ok=2    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
```

实践案例二、所有为web主机名的添加nginx仓库，其余的都跳过添加

when中可以使用的表达式传送门

```
[root@m01 /server/playbook]# cat 09_when_match.yml
- hosts: all
  tasks:
  - name: Add Nginx Yum Repository
    yum_repository:
      name: nginx
      description: Nginx Repository
      enabled: yes
      baseurl:
http://nginx.org/packages/centos/7/$basearch/
      gpgcheck: no
    when: (ansible_hostname is match("web|lb"))




#也可以使用and与or方式
#when: (ansible_hostname is match("web")) or
```

```
#       (ansible_hostname is match("lb"))


#when:  #并且
 - (ansible_hostname is match("web"))
 - (ansible_hostname is match("lb"))
#when: (ansible_hostname is match("web")) and
#       (ansible_hostname is match("lb"))
```

2.执行playbook

```
[root@m01 playbook]# ansible-playbook when_yum.yml

PLAY [all]
**********************************************************
**********************************

TASK [Gathering Facts]
**********************************************************
**********************
ok: [172.16.1.7]
ok: [172.16.1.6]
ok: [172.16.1.8]
ok: [172.16.1.5]

#如果主机名不为web相关，则会跳过该tasks
TASK [Add Nginx Yum Repository]
**********************************************************
*************
skipping: [172.16.1.5]
skipping: [172.16.1.6]
ok: [172.16.1.8]
ok: [172.16.1.7]

PLAY RECAP
**********************************************************
**********************************
172.16.1.5                 : ok=1    changed=0
unreachable=0    failed=0
172.16.1.6                 : ok=1    changed=0
unreachable=0    failed=0
```

```
172.16.1.7                 : ok=2    changed=0
unreachable=0    failed=0
172.16.1.8                 : ok=2    changed=0
unreachable=0    failed=0
```

实践案例三、根据前者命令执行的结果进行判断

1.通过register将命令执行结果保存至变量，然后通过when语句进行判断

```
[root@m01 playbook]# cat when_service.yml
- hosts: web
  tasks:
    - name: Check Httpd Server
      command: systemctl is-active httpd
      ignore_errors: yes
      register: check_httpd

    - name: debug outprint
      debug: var=check_httpd    #通过debug的var输出该变量的所
有内容
                                #msg=" :{{check_httpd}}"

    - name: Httpd Restart #如果check_httpd执行命令结果等于0，
则执行重启httpd，否则跳过
      service: name=httpd state=restarted
      when: check_httpd.rc == 0

#1. Check Httpd Server  检查apache是否正在运行 ,运行状态通过
register 存放在 check_httpd变量中

#2. debug outprint      显示下运行信息check_httpd信息显示出来
显示执行过程.

#3. Httpd Restart       重启apache,条件check_httpd.rc == 0
                        # apache正在运行的时候 重启apache

[root@m01 /server/playbook]# cat
10_reg_debug_when_chk_nginx.yml
- hosts: web
  tasks:
    - name: Check Nginx Server
      command: systemctl is-active nginx
      ignore_errors: yes
      register: check_nginx

    - name: debug outprint
      debug: var=check_nginx

    - name: Httpd Restart
      service: name=nginx state=restarted
      when: check_nginx.rc == 0
```

register的修饰符传送门

2.执行playbook

```
[root@m01 playbook]# ansible-playbook when_service.yml

PLAY [web]
**********************************************************
**********************************

TASK [Gathering Facts]
**********************************************************
**********************
ok: [172.16.1.8]
ok: [172.16.1.7]

TASK [Check Httpd Server]
**********************************************************
*******************
fatal: [172.16.1.8]: FAILED! => {"changed": true, "cmd":
["systemctl", "is-active", "httpd"], "delta":
"0:00:00.023433", "end": "2019-01-31 03:17:23.781113",
"msg": "non-zero return code", "rc": 3, "start": "2019-01-
31 03:17:23.757680", "stderr": "", "stderr_lines": [],
"stdout": "inactive", "stdout_lines": ["inactive"]}
...ignoring
changed: [172.16.1.7]
```

```
TASK [Httpd Restart]
**********************************************************
************************
skipping: [172.16.1.8]
changed: [172.16.1.7]

PLAY RECAP
**********************************************************
**********************************
172.16.1.7                 : ok=4    changed=2
unreachable=0    failed=0
172.16.1.8                 : ok=3    changed=1
unreachable=0    failed=0
```

- when条件语句小结
  - **配合ansible 变量 实现判断(ip,主机名,系统)**
  - 配合register变量 实现判断(if $? ==0 执行xxxx服务)
  - 加入或使用debug 显示执行过程
  - 匹配规则: == ; != ; is match() when中可以使用的表达式传送门

2.Playbook循环语句

有时候我们写playbook的时候发现了很多task都要重复引用某个模块，比如一次启动10个服务，或者一次拷贝10个文件，如果按照传统的写法最少要写10次，这样会显得playbook很臃肿。如果使用循环的方式来编写playbook，这样可以减少重复使用某个模块。

```
#for循环


#for  name    in   清单
for  name    in   nginx   mysql  mariadb   httpd   php-fpm
  .....
do
     systemctl  start   $name
done
```

实践案例一、使用循环启动多个服务

1.在没有使用循环的场景下，启动多个服务需要写多条tasks任务。

```
[root@m01 playbook]# cat loop-service.yml
- hosts: web
  tasks:
    - name: Installed Httpd Mariadb Package
      yum: name=httpd,mariadb state=latest

    - name: Start Httpd Server
      service: name=httpd state=started enabled=yes

    - name: Start Mariadb Server
      service: name=mariadb state=started enabled=yes
```

2.我们将如上的playbook修改为循环的方式，减少重复编写多份tasks

```
[root@m01 playbook]# cat loop-service.yml
- hosts: web
  tasks:
    - name: Installed Httpd Mariadb Package
      yum: name=httpd,mariadb-server state=latest
    - name: Start Httpd Mariadb Server
      service: name={{ item }} state=started enabled=yes
      with_items:
        - httpd
        - mariadb



[root@m01 /server/playbook]# cat
11_loop_start_service.yml
- hosts: web
  tasks:
    - name: restart all services
      systemd: name={{ item }}    state=restarted
      with_items:
      - nginx
      - php-fpm
      - crond
      - sshd
```

3.执行playbook

```
[root@m01 /server/playbook]# ansible-playbook -i hosts
11_loop_start_service.yml

PLAY [web]
**********************************************************
**********************************************************
****

TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.9]
ok: [172.16.1.8]
ok: [172.16.1.7]
ok: [172.16.1.10]
```

```
TASK [restart all services]
************************************************************
**********************************************
failed: [172.16.1.9] (item=nginx) => {"ansible_loop_var":
"item", "changed": false, "item": "nginx", "msg": "Could
not find the requested service nginx: host"}
changed: [172.16.1.8] => (item=nginx)
changed: [172.16.1.10] => (item=nginx)
changed: [172.16.1.7] => (item=nginx)
failed: [172.16.1.9] (item=php-fpm) =>
{"ansible_loop_var": "item", "changed": false, "item":
"php-fpm", "msg": "Could not find the requested service
php-fpm: host"}
failed: [172.16.1.10] (item=php-fpm) =>
{"ansible_loop_var": "item", "changed": false, "item":
"php-fpm", "msg": "Could not find the requested service
php-fpm: host"}
changed: [172.16.1.7] => (item=php-fpm)
changed: [172.16.1.8] => (item=php-fpm)
changed: [172.16.1.9] => (item=crond)
changed: [172.16.1.10] => (item=crond)
changed: [172.16.1.7] => (item=crond)
changed: [172.16.1.8] => (item=crond)
changed: [172.16.1.9] => (item=sshd)
changed: [172.16.1.10] => (item=sshd)
changed: [172.16.1.7] => (item=sshd)
changed: [172.16.1.8] => (item=sshd)

PLAY RECAP
************************************************************
************************************************************
****
172.16.1.10                : ok=1    changed=0
unreachable=0    failed=1    skipped=0    rescued=0
ignored=0
172.16.1.7                 : ok=2    changed=1
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.8                 : ok=2    changed=1
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

```
172.16.1.9                 : ok=1    changed=0
unreachable=0    failed=1    skipped=0    rescued=0
ignored=0
```

实践案例二、定义变量方式循环

1.案例二、使用定义变量方式循环安装软件包。

```
[root@m01 playbook]# cat loop-service-v2.yml
- hosts: web
  tasks:
    - name: Installed Httpd Mariadb Package
      yum: name={{ pack }} state=latest
      vars:
       pack:
         - httpd
         - mariadb-server

[root@m01 /server/playbook]# cat
12_loop_user_define_var.yml
- hosts: web
  tasks:
    - name: yum software
      yum: name={{ pack }}    state=present
      vars:
        pack:
          - nginx
          - tree
```

2.执行playbook

```
[root@m01 playbook]# ansible-playbook loop-service-v2.yml

PLAY [web]
**********************************************************
**********************************
```

```
TASK [Gathering Facts]
***********************************************************
**********************
ok: [172.16.1.8]
ok: [172.16.1.7]

TASK [Installed Httpd Mariadb Package]
***********************************************************
******
ok: [172.16.1.7]
ok: [172.16.1.8]

PLAY RECAP
***********************************************************
**********************************
172.16.1.7                 : ok=2    changed=0
unreachable=0    failed=0
172.16.1.8                 : ok=2    changed=0
unreachable=0    failed=0
```

实践案例三、使用字典循环方式创建用户和批量拷贝文件

1.批量创建用户，使用key values字典的方式

```
useradd   -u 888       oldboy01
useradd   -u 999       lidao
useradd   -u 1111      lidaoav

这里我们需要2个变量存放信息 一个存放用户名,一个存放uid.....

{ name: 'lidao',    uid: '888'}
{ name: 'lidaoav',  uid: '889'}
{ name: 'lidao996', uid: '886'}







name=lidao
uid=888
sex=男
age=18
```

```
{ name: 'lidao',uid: '888',sex: '男',age: 18 }
{ name: 'lidao',uid: '888',sex: '男',age: 18 }
{ name: 'lidao',uid: '888',sex: '男',age: 18 }
{ name: 'lidao',uid: '888',sex: '男',age: 18 }
{ name: 'lidao',uid: '888',sex: '男',age: 18 }
{ name: 'lidao',uid: '888',sex: '男',age: 18 }
{ name: 'lidao',uid: '888',sex: '男',age: 18 }




useradd   -u 888        oldboy01
useradd   -u 999        lidao
useradd   -u 1111       lidaoav



#add user
- hosts: web
  tasks:
    - name: Add Users
      user: name={{ item.name }} uid={{ item.uid }}
state=present
      with_items:
        - { name: 'oldboy01', uid: '888' }
        - { name: 'lidao'    , uid: '999' }



[root@m01 /server/playbook]# cat 13_dict_uesradd.yml
- hosts: all
  tasks:
   - name: uesradd
     user: name={{ item.name }}    uid={{ item.uid }}
state=present
     with_items:
     - { name: 'oldboy01', uid: 888  }
     - { name: 'lidao'   , uid: 999 }
     - { name: 'oldboy996', uid: 1888  }




[root@manager ~]# cat loop-user.yml
```

```
- hosts: web
  tasks:
    - name: Add Users
      user: name={{ item.name }} groups={{ item.groups }}
state=present
      with_items:
        - { name: 'testuser1', groups: 'bin' }
        - { name: 'testuser2', groups: 'root' }
```

2.执行playbook

```
[root@m01 playbook]# ansible-playbook loop-user.yml

PLAY [web]
**********************************************************
**********************************

TASK [Gathering Facts]
**********************************************************
**********************
ok: [172.16.1.8]
ok: [172.16.1.7]

TASK [Add Users]
**********************************************************
****************************
changed: [172.16.1.7] => (item={u'name': u'testuser1',
u'groups': u'bin'})
changed: [172.16.1.8] => (item={u'name': u'testuser1',
u'groups': u'bin'})
changed: [172.16.1.8] => (item={u'name': u'testuser2',
u'groups': u'root'})
changed: [172.16.1.7] => (item={u'name': u'testuser2',
u'groups': u'root'})

PLAY RECAP
**********************************************************
**********************************
172.16.1.7                 : ok=2    changed=1
unreachable=0    failed=0
```

```
172.16.1.8                 : ok=2    changed=1
unreachable=0    failed=0
```

3.批量拷贝文件，使用key values字典的方式

```
[root@manager ~]# cat loop-file.yml
- hosts: all
  tasks:
    - name: Configure Rsync Server
      copy: src={{ item.src }} dest=/etc/{{ item.dest }}
mode={{ item.mode }}
      with_items:
        - {src: "rsyncd.conf", dest: "rsyncd.conf", mode:
"0644"}
        - {src: "rsync.passwd", dest: "rsync.passwd",
mode: "0600"}
```

- 小结循环
  - **with_items 实现单个变量循环**
  - **with_tiems实现多个变量循环(字典)**
  - 通过vars实现自定义的变量循环(了解)

3.Playbook Handlers

Handlers是一个触发器(notify)，也是一个tasks，只不过是一个特殊的tasks，它是需要被tasks触发才会运行。
*只要**配置文件**发生变更，则会触发handlers执行**重启服务**操作，如果配置文件不发生任何变化则不重启。*

应用场景: notify监控配置文件变化， handlers 实现重启服务/重新挂载....

案例一、playbook安装Apache示例

1.安装apache服务playbook

```
[root@m01 ~]# cat webserver.yml
- hosts: web
  remote_user: root
#1.定义变量，在配置文件中调用
  vars:
    http_port: 8881

#2.安装httpd服务
  tasks:
    - name: Install Httpd Server
      yum: name=httpd state=present

#3.使用template模板，引用上面vars定义的变量至配置文件中
    - name: Configure Httpd Server
      template: src=./httpd.conf
dest=/etc/httpd/conf/httpd.conf
      notify:   #调用名称为Restart Httpd Server的handlers(可
以写多个)
        - Restart Httpd Server

#4.启动Httpd服务
    - name: Start Httpd Server
      service: name=httpd state=started enabled=yes

#5.如果配置文件发生变化会调用该handlers下面的对应名称的task
  handlers:
    - name: Restart Httpd Server
      service: name=httpd state=restarted





#批量部署nginx服务
#并且发送nginx配置文件
```

```
#nginx配置文件的端口,可以在剧本中指定.


[root@m01 /server/playbook]# cat 03_nginx.yml
---
- hosts: 172.16.1.9
  vars:
    http_port: 8080
  tasks:
#1.配置nginx yum源
  - name: Add Nginx Yum Repo
    yum_repository:
      name: nginx
      description: nginx repo
      baseurl:
http://nginx.org/packages/centos/$releasever/$basearch/
      enabled: yes
      gpgcheck: yes
      gpgkey: https://nginx.org/keys/nginx_signing.key
#2. 安装nginx
  - name: Install Nginx
    yum:
      name:  nginx
      state: installed
#3. 创建静态页面
  - name: Index FIle
    copy:
      content: "This is ansible website
ansible.oldboy.com"
      dest:  /usr/share/nginx/html/index.html
#4. 推送配置文件并且修改配置文件的内容
  - name: Copy Nginx.d/conf File
    copy:
      src: ./www.conf
      dest: /etc/nginx/conf.d/default.conf
      backup: yes
#5. 监控配置文件,如果发生了变化  重新推送.并且触发 Restart Nginx
动作.
    notify: Restart Nginx
#6. 安装完成后启动nginx
  - name: Start Nginx
```

```
    systemd:
      name: nginx
      state: started
      enabled: yes
#7.配置notify触发后,具体做什么
  handlers:
    - name: Restart Nginx
      systemd:
        name: nginx
        state: reloaded



[root@m01 /server/playbook]# cat  www.conf
server {
  listen {{ http_port }};
  server_name ansible.oldboy.com;
  location  / {
    root /usr/share/nginx/html;
    index index.html;
  }
}
[root@m01 /server/playbook]# cat 14_notify_vars.yml
---
- hosts: web
  vars:
    http_port: 8080
  tasks:
  - name: Add Nginx Yum Repo
    yum_repository:
      name: nginx
      description: nginx repo
      baseurl:
http://nginx.org/packages/centos/$releasever/$basearch/
      enabled: yes
      gpgcheck: yes
      gpgkey: https://nginx.org/keys/nginx_signing.key
  - name: Install Nginx
    yum:
      name:  nginx
      state: installed
  - name: Index FIle
    copy:
      content: "This is ansible website
ansible.oldboy.com"
      dest:  /usr/share/nginx/html/index.html
  - name: Copy Nginx.d/conf File
    copy:
      src: ./www.conf
      dest: /etc/nginx/conf.d/default.conf
      backup: yes
    notify: Restart Nginx
  - name: Start Nginx
    systemd:
      name: nginx
      state: started
      enabled: yes
  handlers:
    - name: Restart Nginx
      systemd:
        name: nginx
        state: reloaded
```

2.只有当我们修改配置文件才会触发handlers

```
[root@m01 playbook]# ansible-playbook webserver.yml

PLAY [web]
**********************************************************
**********************************
```

```
TASK [Gathering Facts]
**********************************************************
**********************
ok: [172.16.1.8]
ok: [172.16.1.7]

TASK [Install Httpd Server]
**********************************************************
*****************
ok: [172.16.1.8]
ok: [172.16.1.7]

TASK [Configure Httpd Server]
**********************************************************
***************
changed: [172.16.1.8]
changed: [172.16.1.7]

TASK [Start Httpd Server]
**********************************************************
*******************
ok: [172.16.1.8]
ok: [172.16.1.7]

RUNNING HANDLER [Restart Httpd Server]
**********************************************************
******
changed: [172.16.1.8]
changed: [172.16.1.7]

PLAY RECAP
**********************************************************
**********************************
172.16.1.7                 : ok=5    changed=2
unreachable=0    failed=0
172.16.1.8                 : ok=5    changed=2
unreachable=0    failed=0
```

3.handlers注意事项
1.无论多少个task通知了相同的handlers，handlers仅会在所有tasks结束后运行一次。
*2.只有task发生**改变**了才会通知handlers，没有改变则不会触发handlers*
3.不能使用handlers替代tasks

4. handers/小结:

- **核心应用场景**: 使用notify+handlers实现文件/配置文件,更新并重启/重新挂载/重新.....
- handlers传送门

4.Playbook任务标签

默认情况下，Ansible在执行一个playbook时，会执行playbook中定义的所有任务。Ansible的标签(Tags)功能可以给单独任务甚至整个playbook打上标签，然后利用这些标签来指定要运行playbook中的个别任务，或不执行指定的任务。

- **一般应用场景: 用于调试,**
  - **运行指定的task**
  - **排除指定的task**

1.打标签的方式有几种，比如:
对一个task打一个标签、对一个task打多个标签、对多个task打一个标签

2、对task打完标签应该如何使用
-t: 执行指定的tag标签任务
--skip-tags: 执行--skip-tags之外的标签任务

案例一、使用-t指定tags执行

1.编写playbook

```
[root@manager ~]# cat nfs.yml
---
- hosts: nfs
  remote_user: root
  tasks:
    - name: Install Nfs Server
      yum: name=nfs-utils state=present
      tags:
        - install_nfs
        - install_nfs-server

    - name: Service Nfs Server
      service: name=nfs-server state=started enabled=yes
      tags: start_nfs-server
```

2.执行playbook

```
[root@m01 /server/playbook]# ansible-playbook -i hosts
15_tags_nfs.yml -t install_nfs

PLAY [172.16.1.9]
**********************************************************
*******************************************************

TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.9]

TASK [Install Nfs Server]
**********************************************************
***********************************************
ok: [172.16.1.9]

PLAY RECAP
**********************************************************
**********************************************************
****
```

```
172.16.1.9                 : ok=2    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0

[root@m01 /server/playbook]# ansible-playbook -i hosts
15_tags_nfs.yml -t install_nfs,start_nfs-server

PLAY [172.16.1.9]
**********************************************************
*******************************************************

TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.9]

TASK [Install Nfs Server]
**********************************************************
***********************************************
ok: [172.16.1.9]

TASK [Service Nfs Server]
**********************************************************
***********************************************
changed: [172.16.1.9]

PLAY RECAP
**********************************************************
**********************************************************
****
172.16.1.9                 : ok=3    changed=1
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

3.使用-t指定tags执行, 多个tags使用逗号隔开即可

```
[root@m01 /server/playbook]# ansible-playbook -i hosts
15_tags_nfs.yml -t install_nfs,start_nfs-server
```

```
PLAY [172.16.1.9]
**********************************************************
*******************************************************

TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.9]

TASK [Install Nfs Server]
**********************************************************
***********************************************
ok: [172.16.1.9]

TASK [Service Nfs Server]
**********************************************************
***********************************************
changed: [172.16.1.9]

PLAY RECAP
**********************************************************
**********************************************************
****
172.16.1.9                 : ok=3    changed=1
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

案例二、使用--skip-tags排除不执行的tags

```
[root@manager ~]# ansible-playbook --skip-tags
install_nfs-server nfs.yml

PLAY [all]
**********************************************************
**********************************************************
************


TASK [Gathering Facts]
**********************************************************
**********************************************************
ok: [172.16.1.31]

TASK [Service Nfs Server]
**********************************************************
*******************************************************
ok: [172.16.1.31]

PLAY RECAP
**********************************************************
**********************************************************
************
172.16.1.31                  : ok=2    changed=0
unreachable=0    failed=0
```

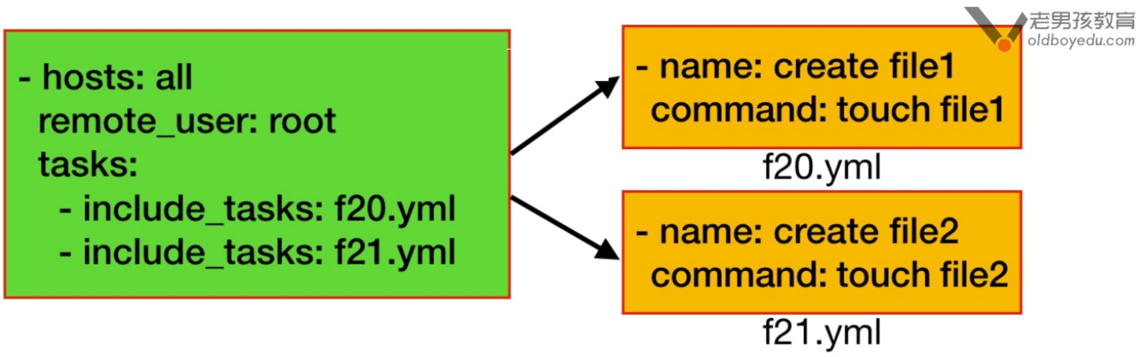
- 应用案例: 运行nginx自动化管理剧本指定的task

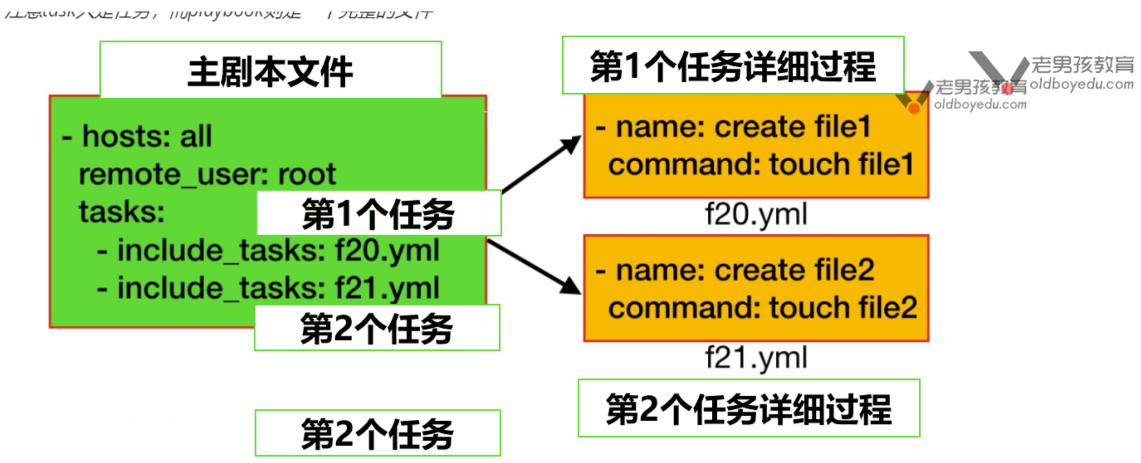
```
[root@m01 /server/playbook]# cat 14_auto_nginx.yml
---
- hosts: web
  vars:
    http_port: 1234
  tasks:
  - name: Add Nginx Yum Repo
    yum_repository:
      name: nginx
      description: nginx repo
      baseurl:
http://nginx.org/packages/centos/$releasever/$basearch/
      enabled: yes
      gpgcheck: yes
      gpgkey: https://nginx.org/keys/nginx_signing.key
  - name: Install Nginx
    yum:
      name:  nginx
      state: installed
  - name: Index FIle
    copy:
      content: "This is ansible website
ansible.oldboy.com"
      dest:  /usr/share/nginx/html/index.html
  - name: Copy Nginx.d/conf File
    template:
      src: ./www.conf
      dest: /etc/nginx/conf.d/default.conf
      backup: yes
    tags:
    - push_config_file
    notify: Restart Nginx
  - name: Start Nginx
    systemd:
      name: nginx
      state: started
      enabled: yes
  handlers:
    - name: Restart Nginx
      systemd:
        name: nginx
        state: reloaded
[root@m01 /server/playbook]# ansible-playbook -i hosts
14_auto_nginx.yml  -t push_config_file

PLAY [web]
**********************************************************
**********************************************************
****

TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.8]
ok: [172.16.1.9]
```

```
ok: [172.16.1.7]
ok: [172.16.1.10]

TASK [Copy Nginx.d/conf File]
**********************************************************
*******************************************
changed: [172.16.1.8]
changed: [172.16.1.10]
changed: [172.16.1.9]
changed: [172.16.1.7]

RUNNING HANDLER [Restart Nginx]
**********************************************************
*****************************************
changed: [172.16.1.7]
changed: [172.16.1.8]
changed: [172.16.1.9]
changed: [172.16.1.10]

PLAY RECAP
**********************************************************
**********************************************************
****
172.16.1.10                : ok=3    changed=2
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.7                 : ok=3    changed=2
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.8                 : ok=3    changed=2
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.9                 : ok=3    changed=2
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

5.Playbook文件复用

include_tasks用来动态的包含tasks任务文件，当然也可以使用import_playbook导入playbook文件
注意task只是任务，而playbook则是一个完整的文件





include调用任务方式

```
#主入口文件
[root@mha ~]# cat main.yml
- hosts: all
  remote_user: root
  tasks:
    - include_tasks: f20.yml
    - include_tasks: f21.yml

#f20.yml
[root@mha ~]# cat f20.yml
- name: create file1
  command: touch file1

#21.yml
[root@mha ~]# cat f21.yml
- name: create file2
  command: touch file2


[root@m01 /server/playbook]# cat main.yml
- hosts: all
  remote_user: root
  tasks:
    - include_tasks: 01-basic.yml
    - include_tasks: 02-install-web.yml
[root@m01 /server/playbook]# cat 01-basic.yml
- name: basic  youhua
  debug:
    msg: "this is  basic youhua............"
[root@m01 /server/playbook]# cat 02-install-web.yml
- name: install web servers
  debug:
    msg:
    - "installing  web servers  nginx..............."
    - "installing  web servers  php..............."
    - "installing  web servers  nfs..............."
  when: (ansible_hostname is  match("web"))
[root@m01 /server/playbook]# ansible-playbook -i hosts
[root@m01 /server/playbook]# ansible-playbook -i hosts
main.yml  -C

PLAY [all]
**********************************************************
**********************************************************
****
```

```
TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.41]
ok: [172.16.1.9]
ok: [172.16.1.31]
ok: [172.16.1.51]
ok: [172.16.1.42]
ok: [172.16.1.5]
ok: [172.16.1.6]
ok: [172.16.1.8]
ok: [172.16.1.7]
ok: [172.16.1.10]

TASK [include_tasks]
**********************************************************
****************************************************
included: /server/playbook/01-basic.yml for 172.16.1.9,
172.16.1.42, 172.16.1.31, 172.16.1.41, 172.16.1.51,
172.16.1.5, 172.16.1.6, 172.16.1.7, 172.16.1.8,
172.16.1.10

TASK [basic  youhua]
**********************************************************
****************************************************
ok: [172.16.1.9] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.42] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.31] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.41] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.51] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.5] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.6] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.7] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.8] => {
    "msg": "this is  basic youhua..........."
}
ok: [172.16.1.10] => {
    "msg": "this is  basic youhua..........."
}

TASK [include_tasks]
***********************************************************
*****************************************************
included: /server/playbook/02-install-web.yml for
172.16.1.9, 172.16.1.42, 172.16.1.31, 172.16.1.41,
172.16.1.51, 172.16.1.5, 172.16.1.6, 172.16.1.7,
172.16.1.8, 172.16.1.10

TASK [install web servers]
***********************************************************
***********************************************
ok: [172.16.1.9] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}
skipping: [172.16.1.42]
skipping: [172.16.1.31]
skipping: [172.16.1.41]
skipping: [172.16.1.51]
skipping: [172.16.1.5]
skipping: [172.16.1.6]
ok: [172.16.1.7] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}
ok: [172.16.1.8] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}
ok: [172.16.1.10] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}

PLAY RECAP
**********************************************************
**********************************************************
****
172.16.1.10                : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.31                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.41                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.42                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.5                 : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
```

```
172.16.1.51                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.6                 : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.7                 : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.8                 : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.9                 : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0

[root@m01 /server/playbook]#
[root@m01 /server/playbook]# ansible-playbook -i hosts
main.yml

PLAY [all]
**********************************************************
**********************************************************
****

TASK [Gathering Facts]
**********************************************************
**************************************************
ok: [172.16.1.41]
ok: [172.16.1.9]
ok: [172.16.1.51]
ok: [172.16.1.31]
ok: [172.16.1.42]
ok: [172.16.1.5]
ok: [172.16.1.7]
ok: [172.16.1.6]
ok: [172.16.1.8]
ok: [172.16.1.10]
```

```
TASK [include_tasks]
**********************************************************
****************************************************
included: /server/playbook/01-basic.yml for 172.16.1.9,
172.16.1.42, 172.16.1.31, 172.16.1.41, 172.16.1.51,
172.16.1.5, 172.16.1.6, 172.16.1.7, 172.16.1.8,
172.16.1.10

TASK [basic  youhua]
**********************************************************
****************************************************
ok: [172.16.1.9] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.42] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.31] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.41] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.51] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.5] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.6] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.7] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.8] => {
    "msg": "this is  basic youhua............"
}
ok: [172.16.1.10] => {
    "msg": "this is  basic youhua............"
}
```

```
TASK [include_tasks]
**********************************************************
****************************************************
included: /server/playbook/02-install-web.yml for
172.16.1.9, 172.16.1.42, 172.16.1.31, 172.16.1.41,
172.16.1.51, 172.16.1.5, 172.16.1.6, 172.16.1.7,
172.16.1.8, 172.16.1.10

TASK [install web servers]
**********************************************************
**********************************************
ok: [172.16.1.9] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}
skipping: [172.16.1.42]
skipping: [172.16.1.31]
skipping: [172.16.1.41]
skipping: [172.16.1.51]
skipping: [172.16.1.5]
skipping: [172.16.1.6]
ok: [172.16.1.7] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}
ok: [172.16.1.8] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}
ok: [172.16.1.10] => {
    "msg": [
        "installing  web servers  nginx...............",
        "installing  web servers  php...............",
        "installing  web servers  nfs..............."
    ]
}

PLAY RECAP
************************************************************
************************************************************
****
172.16.1.10                : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.31                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.41                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.42                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.5                 : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.51                : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.6                 : ok=4    changed=0
unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.16.1.7                 : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.8                 : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.16.1.9                 : ok=5    changed=0
unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

```
[root@m01 /server/playbook]#
```

6.Playbook忽略错误

*默认Playbook会检tasks执行的**返回状态(rc)**，如遇到错误则(rc!=0) 会立即终止playbook的后续的tasks执行。*

*然而有些时候palybook即使执行错误了也要让其继续执行. ***
加入参数: ignore_errors: yes 忽略错误

1.编写playbook，当有task执行失败则会立即终止后续task运行

```
[root@manager ~]# cat f9.yml
---
- hosts: all
  remote_user: root
  tasks:
    - name: Ignore False
      command: /bin/false
      ignore_errors: yes

    - name: touch new file
      file: path=/tmp/oldboy_ignore state=touch
```

2.执行playbook，会发现报错了，后续的任务也没有进行执行。

```
[root@m01 playbook]# ansible-playbook ignore.yml

PLAY [web]
**********************************************************
**********************************

TASK [Gathering Facts]
**********************************************************
**********************
ok: [172.16.1.7]
```

```
ok: [172.16.1.8]

TASK [Ignore False]
**********************************************************
*************************
fatal: [172.16.1.8]: FAILED! => {"changed": true, "cmd":
["/bin/false"], "delta": "0:00:00.021502", "end": "2019-
01-31 20:27:51.206530", "msg": "non-zero return code",
"rc": 1, "start": "2019-01-31 20:27:51.185028", "stderr":
"", "stderr_lines": [], "stdout": "", "stdout_lines": []}
fatal: [172.16.1.7]: FAILED! => {"changed": true, "cmd":
["/bin/false"], "delta": "0:00:00.022049", "end": "2019-
01-31 20:27:51.206340", "msg": "non-zero return code",
"rc": 1, "start": "2019-01-31 20:27:51.184291", "stderr":
"", "stderr_lines": [], "stdout": "", "stdout_lines": []}
    to retry, use: --limit
@/etc/ansible/playbook/ignore.retry

PLAY RECAP
**********************************************************
**********************************
172.16.1.7                 : ok=1    changed=0
unreachable=0    failed=1
172.16.1.8                 : ok=1    changed=0
unreachable=0    failed=1
```

3.我们可以给对应的task任务添加忽略错误

```
[root@m01 playbook]# cat ignore.yml
- hosts: web
  tasks:
    - name: Ignore False
      command: /bin/false #该命令会返回非0,代表命令执行失败
      ignore_errors: yes  #忽略错误

    - name: touch new file
      file: path=/tmp/oldboy_ignore state=touch
```

4.再次执行playbook,如果碰到tasks错误，会自动忽略，继续执行剩下的tasks

```
[root@m01 playbook]# ansible-playbook ignore.yml

PLAY [web]
**********************************************************
**********************************

TASK [Gathering Facts]
**********************************************************
**********************
ok: [172.16.1.8]
ok: [172.16.1.7]

TASK [Ignore False]
**********************************************************
*************************
fatal: [172.16.1.7]: FAILED! => {"changed": true, "cmd":
["/bin/false"], "delta": "0:00:00.019128", "end": "2019-
01-31 20:30:45.710746", "msg": "non-zero return code",
"rc": 1, "start": "2019-01-31 20:30:45.691618", "stderr":
"", "stderr_lines": [], "stdout": "", "stdout_lines": []}
...ignoring
fatal: [172.16.1.8]: FAILED! => {"changed": true, "cmd":
["/bin/false"], "delta": "0:00:00.020302", "end": "2019-
01-31 20:30:45.715142", "msg": "non-zero return code",
"rc": 1, "start": "2019-01-31 20:30:45.694840", "stderr":
"", "stderr_lines": [], "stdout": "", "stdout_lines": []}
...ignoring

TASK [touch new file]
**********************************************************
***********************
changed: [172.16.1.8]
changed: [172.16.1.7]

PLAY RECAP
**********************************************************
**********************************
172.16.1.7                 : ok=3    changed=2
unreachable=0    failed=0
172.16.1.8                 : ok=3    changed=2
unreachable=0    failed=0
```