



# TENSOR

## Введение в администрирование Linux: Ansible Семинар #3 “Ansible: Написание сценариев”

Пантелеев Александр



# Введение в администрирование Linux: Ansible

## План занятия

- Блоки
- Тэги
- Статические и динамические вызовы
- Роли

*Вопросы просьба задавать в конце подтемы.*

# Блоки

Выражение block реализует механизм группировки задач.

Выражение block позволяет определять условия и опции сразу для всех задач в блоке

```
1  - name: block example
2    hosts: nodes
3    gather_facts: true
4    become: true
5    tasks:
6      - name: nginx start on centos systems
7        block:
8          - name: set fact os distribution
9            set_fact:
10              os_distribution: "{{ ansible_facts.distribution }}"
11
12          - name: set fact os distribution version
13            set_fact:
14              os_full_version: "{{ ansible_facts.distribution_version }}"
15
16          - name: print alltogether
17            debug:
18              msg: "Current host's OS version is {{ os_distribution }} {{ os_full_version }}"
19          when: ansible_facts.distribution | lower == 'centos'
20
```

# Обработка ошибок с помощью блоков

- Реализация обработки ошибок в Ansible напоминает парадигму try-catch-finally и работает похожим образом.
- Задачи из `always` выполняются всегда. Даже после обнаружения ошибок выполнения задач в выражении `rescue`.

```
1 - name: get facts playbook
2   hosts: localhost
3   connection: local
4   gather_facts: false
5   tasks:
6     - name: error handling block
7       block:
8         - name: debug
9           debug:
10             msg: "You will see failed tasks after this"
11
12         - name: always failed
13           command: /bin/false
14
15         - name: unreachable debug
16           debug:
17             msg: "You won't see this message"
18       rescue:
19         - name: emergency debug
20           debug:
21             msg: "This task will be executed in case of block tasks failure"
22
23         - name: also failed
24           command: /bin/false
25       always:
26         - name: final debug
27           debug:
28             msg: "This task will always be executed"
29
```

# Повышение привилегий для блоков

Ansible предоставляет возможность повышать привилегии пользователя для отдельных блоков задач.

```
1 ---
2 - name: Deploy webserver
3   hosts: nodes
4   become: false
5   tasks:
6     - name: Block install and start nginx with root privileges
7       become: true
8       block:
9         - name: ensure nginx repo files exist on remote host
10           copy:
11             src: nginx.repo
12             dest: /etc/yum.repos.d/nginx.repo
13             owner: root
14             group: root
15             mode: 0644
16         - name: Ensure nginx is installed
17           yum:
18             name: nginx
19             state: present
20         - name: Ensure nginx is started and enabled
21           service:
22             name: nginx
23             state: started
24             enabled: true
25         - name: Populate service facts
26           service_facts:
27             register: service_facts_list
28         - name: Ensure firewalld rule for nginx is enabled
29           firewalld:
30             port: 80/tcp
31             permanent: true
32             state: enabled
33             immediate: true
34             when: service_facts_list.ansible_facts.services["firewalld.service"].state == "running"
35         # блок заканчивается здесь
36     - name: Test connection without root privileges
37       uri:
38         url: "http://{{ ansible_host }}"
39         return_content: true
40       register: webpage
41       failed_when: webpage.status != 200
42       connection: local
43
```

# Тэги

Ansible позволяет применять тэги для выполнения определенных сценариев или задач.

Для добавления тэгов к ресурсам необходимо использовать ключевое слово `tags`, за которым следует список нужных тегов.

Тэги доступны для следующих ресурсов.

- Задачи
- Операции
- Блоки задач
- Роли

Запуск сценария с передачей тэга:

```
$ ansible-playbook 01_job_tags.yaml --tags=prod
```

Пропустить задачи с определенным тэгом:

```
$ ansible-playbook 01_job_tags.yaml --skip-tags=test
```

Проверить доступные тэги:

```
$ ansible-playbook 01_job_tags.yaml --list-tags
```

[Ссылка на документацию](#)

```
1 ---
2 - name: job tags
3   hosts: nodes
4   become: true
5   tasks:
6     - name: task with prod tag
7       debug:
8         msg: "prod"
9       tags:
10        - prod
11
12    - name: task with test tag
13      debug:
14        msg: "test"
15      tags:
16        - test
17
```

```
1 ---
2 - name: 1. play tags - prod
3   hosts: nodes
4   tags:
5     - prod
6   tasks:
7     - name: 1. task from play with prod tag
8       debug:
9         msg: "prod"
10
11 - name: 2. play tags - test
12   hosts: nodes
13   tags:
14     - test
15   tasks:
16     - name: 2. task from play with test tag
17       debug:
18         msg: "test"
19
```

# Тэги + блоки

```
1 ---
2 - name: Setup nginx webserver
3   hosts: nodes
4   tasks:
5     - name: install and start nginx
6       block:
7         - name: Ensure nginx is installed
8           yum:
9             name: nginx
10            state: present
11        - name: Ensure nginx is started and enabled
12          service:
13            name: nginx
14            state: started
15            enabled: true
16        - name: Populate service facts
17          service_facts:
18            register: service_facts_list
19        - name: Ensure firewall rule for nginx is enabled
20          firewallld:
21            port: 80/tcp
22            permanent: true
23            state: enabled
24            immediate: true
25            when: service_facts_list.ansible_facts.services["firewalld.service"].state == "running"
26          # Тег относится ко всему блоку
27          become: true
28          tags:
29            - webserver
30
31        - name: Test connection without root priveleges
32          uri:
33            url: "http://{{ ansible_host }}"
34            return_content: true
35            register: webpage
36            failed_when: webpage.status != 200
37            connection: local
38            tags:
39              - webserver
40              - check
41
```

# Специальные тэги

В Ansible предусмотрен специальный тэг, который можно назначать в наборе сценариев — `always`. Ресурс, помеченный тэгом `always`, всегда будет выполняться, даже если он не соответствует списку тэгов, переданных в опции `--tags`. Единственное исключение — когда он явно пропускается с помощью опции `--skip-tags always`.

Задача, помеченная тэгом `never`, не выполняется, за исключением случая, когда выполняется набор сценариев с опцией `--tags`, в которой указано значение `never` или один из тэгов, связанных с задачей.

Есть еще три специальных тэга.

- Тэг `tagged` запускает любой ресурс с явным тэгом.
- Тэг `untagged` запускает любой ресурс, не имеющий явного тэга, и исключает все помеченные тэгами ресурсы.
- Тэг `all` включает все задачи в сценарии независимо от того, есть у них тэги или нет. Это поведение Ansible по умолчанию.



# Задачи: Import

```
1 ---
2 - name: execute tasks depending on OS version
3   gather_facts: true
4   hosts: nodes
5   tasks:
6     - name: set os_distrib_version fact
7       set_fact:
8         os_distrib: "{{ ansible_facts.distribution }}"
9     - name:
10       debug:
11         var: os_distrib
12     - name: set os_distrib_version fact
13       set_fact:
14         os_distrib_version: "{{ ansible_facts.distribution_version }}"
15     - name:
16       debug:
17         var: os_distrib_version
18     - name: execute tasks file dynamically
19       import_tasks: centos7.9.yaml
20       when:
21         - os_distrib | lower == "centos"
22         - os_distrib_version | string == "7.9"
23
```

```
1 - name:
2   debug:
3     msg: "Tasks file imported successfully"
4
5 - name: set os_distrib_version fact
6   set_fact:
7     os_distrib: "BolgenOS"
8
9 - name: conditional expression check (skipped)
10  debug:
11    msg: "os_distrib conditional check on import_tasks is applied to this task"
12
```

[Ссылка на документацию](#)

# Задачи: Include

```
1 ---
2 - name: execute tasks depending on OS version
3   gather_facts: true
4   hosts: nodes
5   tasks:
6     - name: set os_distrib_version fact
7       set_fact:
8         os_distrib: "{{ ansible_facts.distribution | lower }}"
9     - name:
10       debug:
11         var: os_distrib
12     - name: set os_distrib_version fact
13       set_fact:
14         os_distrib_version: "{{ ansible_facts.distribution_version }}"
15     - name:
16       debug:
17         var: os_distrib_version
18     - name: execute tasks file dynamically
19       include_tasks: "{{ os_distrib }}{{ os_distrib_version }}.yaml"
20       when:
21         - os_distrib | lower == "centos"
22         - os_distrib_version | string == "7.9"
23
```

```
1 - name:
2   debug:
3     msg: "Tasks file included successfully"
4
5 - name: set os_distrib_version fact
6   set_fact:
7     os_distrib: "BolgenOS"
8
9 - name: conditional expression check
10  debug:
11    msg: "os_distrib conditional check on include_tasks is not applied to this task"
12
```

[Ссылка на документацию](#)

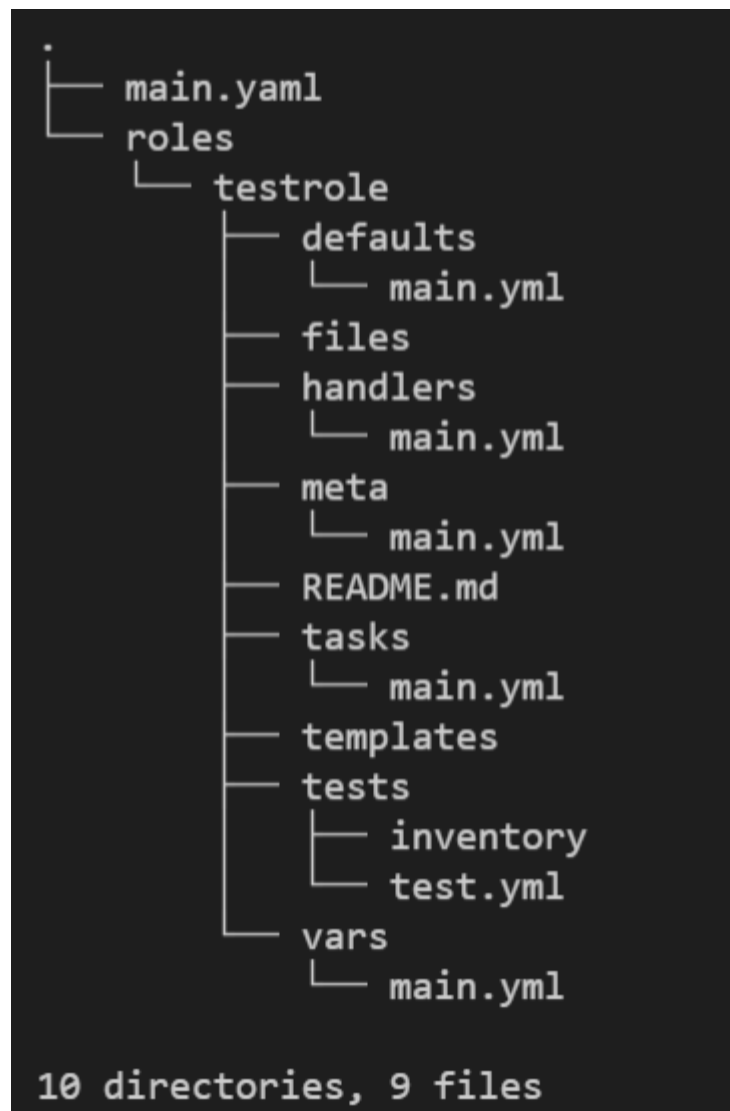
# Статический и динамический вызов

Состояние	Import (Статический)	Include (Динамический)
Проверка	Ansible проверит синтаксис задач внутри импортированного файла перед запуском сценария	Ansible проверит синтаксис задач внутри файла только в момент вызова задачи с модулем <code>include_tasks</code> .
Условные выражения	Условие отдельно проверяется для каждой задачи внутри импортированного файла	Условие проверяется только для вызова файла с задачами Можно применить проверку условия ко всем задачам, если указать её через опцию <code>apply</code>
Тэги	Если тэг применен к <code>import_tasks</code> , то выполнятся все задачи внутри импортированного файла	Если тэг применен к <code>include_tasks</code> , выполнятся только те задачи внутри файла, на которые поставлен тэг. Можно применить тэги ко всем задачам, если указать его через опцию <code>apply</code>
Повышение привилегий	Если повышение привилегий указано как ключевое слово на задаче с импортом, то повышение привилегий применится ко всем задачам внутри импортированного файла.	Повышение привилегий применяется ко всем задачам только если указать повышение привилегий через опцию <code>apply</code>

# Роли

Роль в Ansible — основной механизм деления сценария на отдельные файлы. Они упрощают написание сценариев и их повторное использование.

```
$ ansible-galaxy init testrole
```



# roles

Роли вызываются через выражение roles, в которое передается список имен ролей

## Список выполнения:

1. pre\_tasks сценария
2. Обработчики, вызываемые в pre\_tasks.
3. Все роли, перечисленные в выражении roles по порядку. При этом сначала выполняются все зависимые роли, записанные в meta файле.
4. Остальные задачи, определенные в сценарии
5. Обработчики, вызываемые ролями или задачами
6. post\_tasks сценария
7. Обработчики, вызываемые из post\_tasks

Этот механизм приносит некоторые неудобства: недостаточно гибкости для определения порядка, в котором будут выполняться роли и задачи.

Статический механизм вызова ролей.

```
1 ---
2 - name: using roles
3   hosts: localhost
4   connection: local
5   vars:
6     test_part: test
7   pre_tasks:
8     - name: pre-task test
9       debug:
10        msg: "This task executes before roles' tasks"
11      # - name: set fact
12      #   set_fact:
13      #     test_part: test
14   roles:
15     - role: "{{ test_part }}role"
16       vars:
17         custom_var: 42
18         tags: tagA
19         when: test_part == "test"
20   tasks:
21     - name: task from main playbook
22       debug:
23        msg: "This task executes after roles' tasks"
24
```

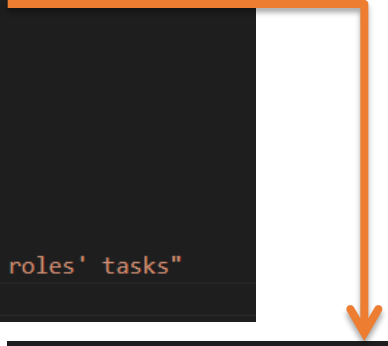
```
1 ---
2 # tasks file for testrole
3 - name: debug test (no tag applied)
4   debug:
5     msg: "this task executes in role"
6
7 - name: debug external var (tagA applied)
8   debug:
9     msg: "{{ custom_var }}"
10  tags:
11    - tagA
12
```



# include\_role

- Динамический вызов роли

```
1 ---
2 - name: include_role tasks
3   hosts: localhost
4   connection: local
5   vars:
6     test_part: test
7   # pre_tasks:
8   #   - name: set fact
9   #     set_fact:
10   #       test_part: test
11   tasks:
12     - name: task before role
13       debug:
14         msg: "This task executes before roles' tasks"
15
16     - name: dynamic role include_role
17       include_role:
18         name: "{{ test_part }}role"
19       vars:
20         custom_var: 42
21         tags: tagA
22         when: test_part == "test"
23
24     - name: task after role
25       debug:
26         msg: "This task executes after roles' tasks"
27
```



```
1 ---
2 # tasks file for testrole
3 - name: debug test (no tag applied)
4   debug:
5     msg: "this task executes in role"
6
7 - name: debug external var (tagA applied)
8   debug:
9     msg: "{{ custom_var }}"
10   tags: tagA
11
```

# import\_role

- Статический вызов роли

```
1 ---
2 - name: include_role tasks
3   hosts: localhost
4   connection: local
5   vars:
6     test_part: test
7   # pre_tasks:
8   #   - name: set fact
9   #     set_fact:
10   #       test_part: test
11   tasks:
12     - name: task before role
13       debug:
14         msg: "This task executes before roles' tasks"
15
16     - name: static role import_role
17       import_role:
18         name: "{{ test_part }}role"
19       vars:
20         custom_var: 42
21       tags: tagA
22       when: test_part == "test"
23
24     - name: task after role
25       debug:
26         msg: "This task executes after roles' tasks"
```

```
1 ---
2 # tasks file for testrole
3 - name: debug test
4   debug:
5     msg: "this task executes in role"
6
7 - name: debug external var
8   debug:
9     msg: "{{ custom_var }}"
10  tags: tagA
11
```

# Домашнее задание

Переписать сценарии установки и настройки Nginx и MariaDB под использование ролей.

1 роль: Создание сертификатов

2 роль: Установка и конфигурация Nginx, копирование файлов с содержимым веб-сайта, применение правил firewalld для http/https-порта, активация службы Nginx, обработчики на изменение конфигурационных файлов и файлов с содержимым.

3 роль: Установка MariaDB, активация службы, создание базы данных и роли пользователя.

Дополнительный балл, если внедрите возможность разворачивать только MariaDB или только Nginx при передаче соответствующего тэга



Познакомились с блоками,  
тэгами, ролями.  
Вопросы?

