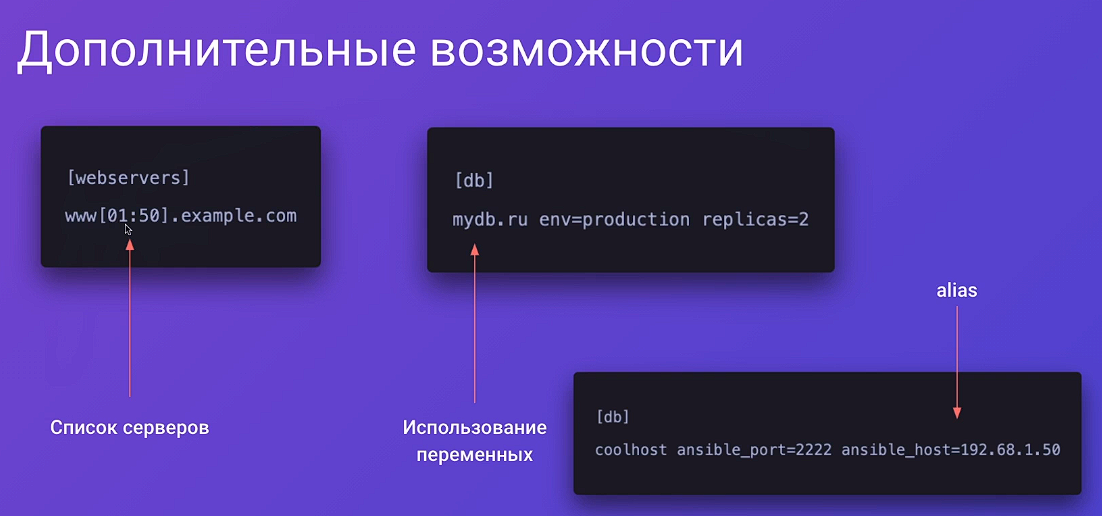
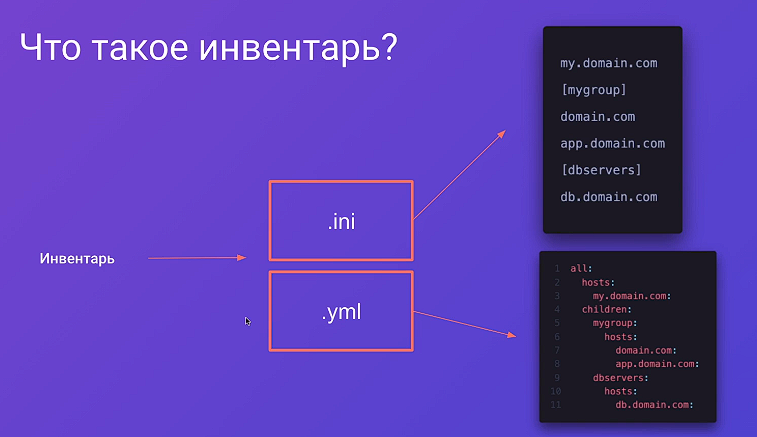
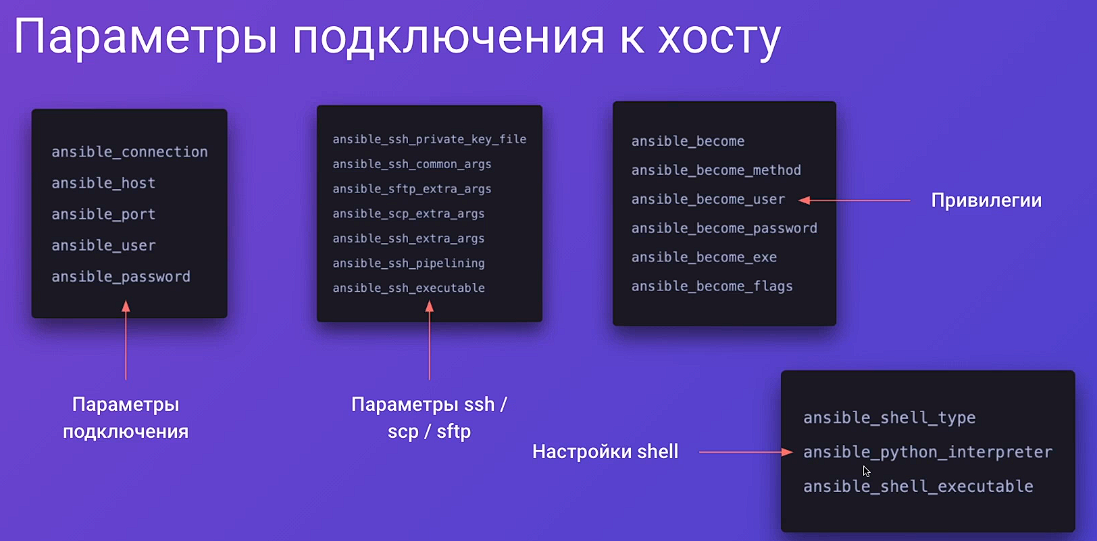
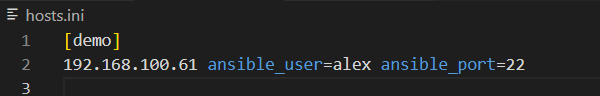
**Ansible**

# Inventory:

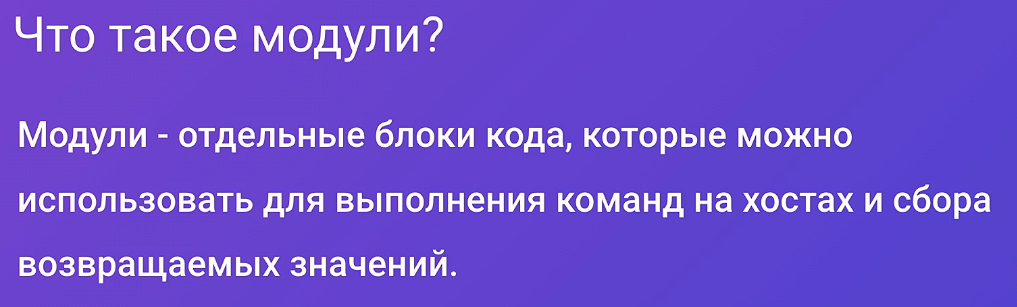


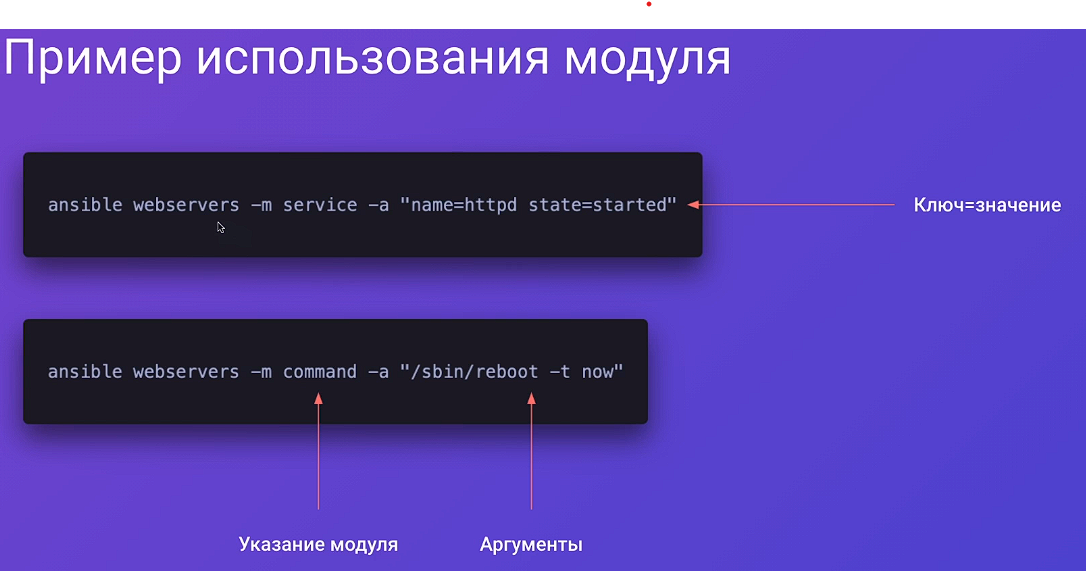




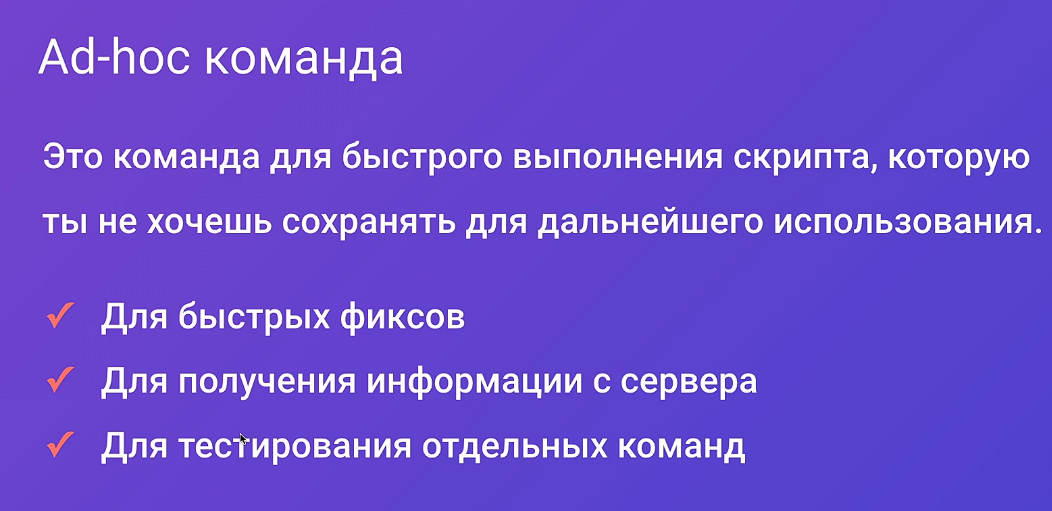
* ansible -i hosts.ini -m ping demo --- -i /запуск инвентаря, -m /модуль , demo- название модуля в инвентаре;

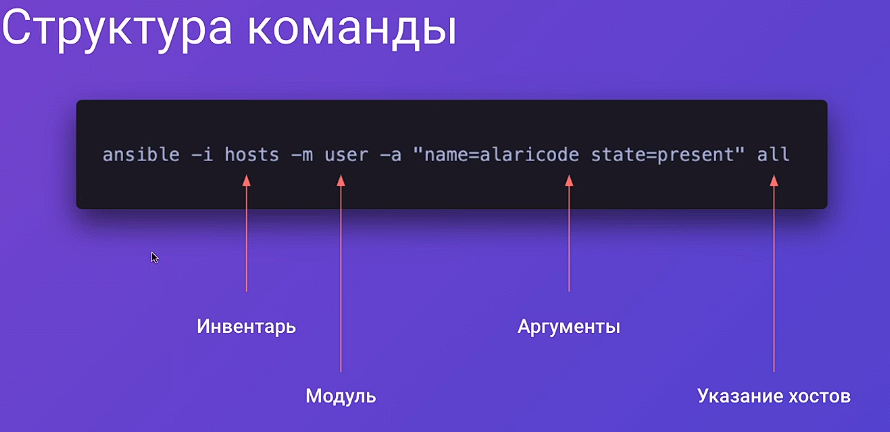
# Модули

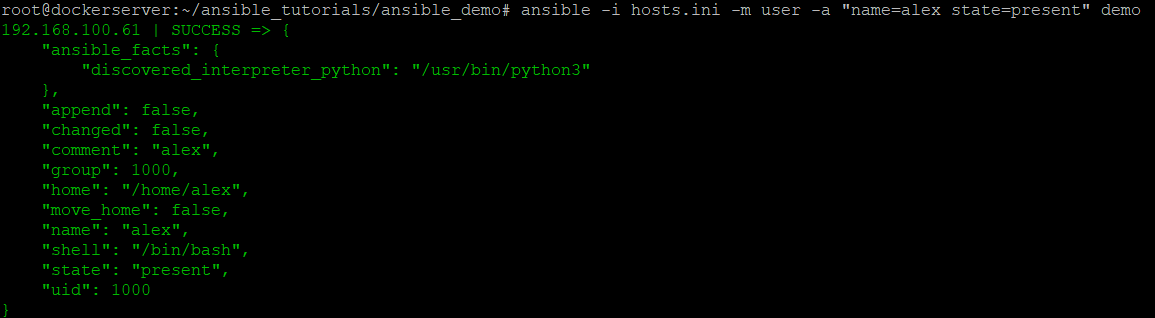




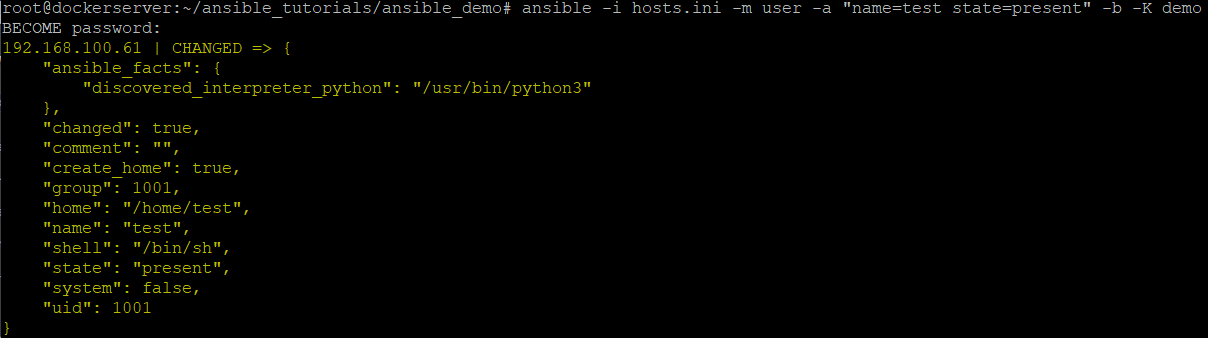
# Ad-hoc команды







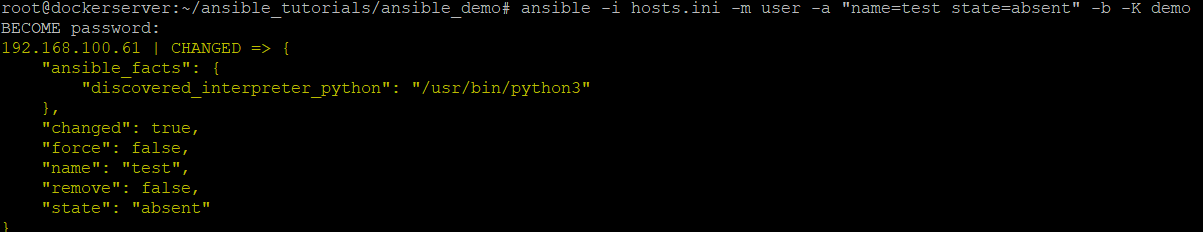
--- Проверяем, если существует пользователь на сервере;



--- **-b = BECOME** /запускаем команды из под sudo юзера;

--- **-K = ask password** /запрашиваем пароль от sudo;

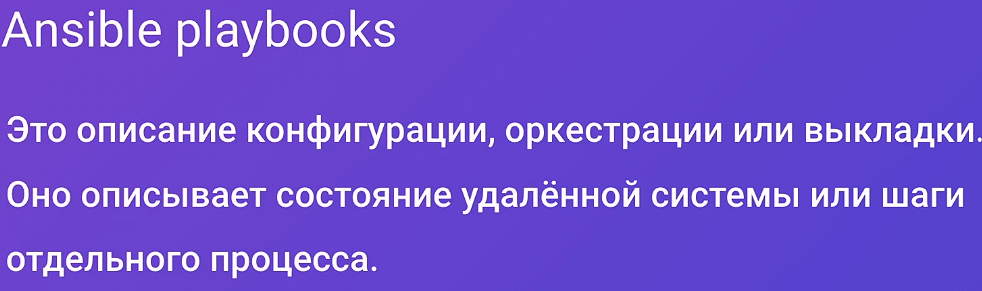
--- **name=test state=present** /добавляем нового пользователя test;

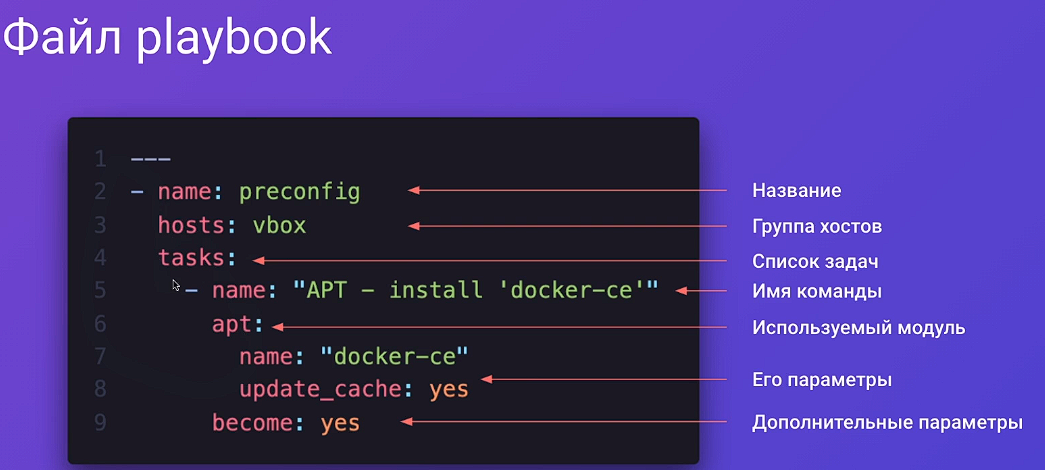


--- **name=test state=absent** /удаляем пользователя test;

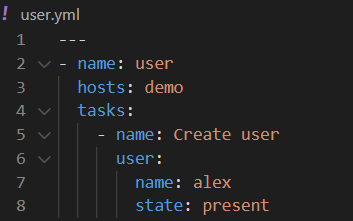
* **ansible -i hosts –m user –a “name=my\_user state=present” my\_group** --- создаём пользователя my\_user с помощью ad-hoc команды на группе серверов my\_group в инвентаре hosts.

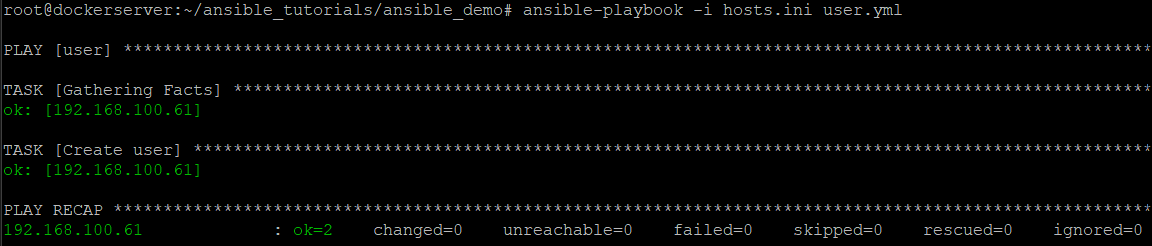
# Playbook



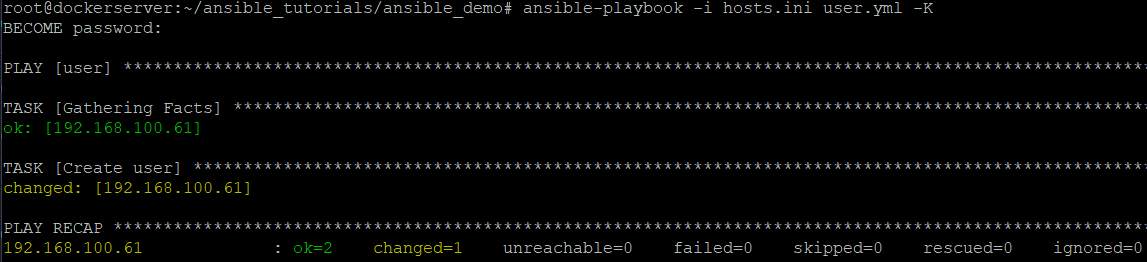


Пример:

 Создаем (проверяем наличие пользователя в данном случае) ;

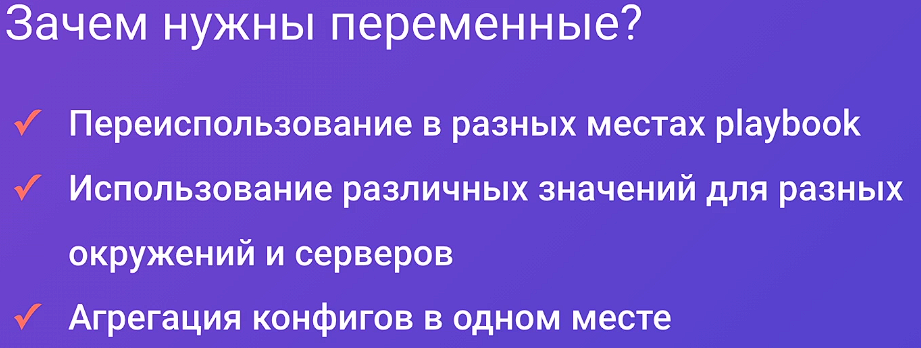


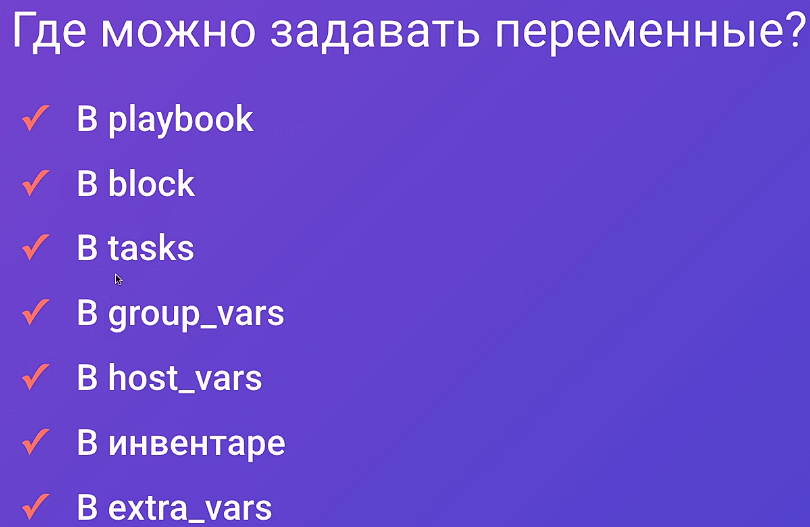
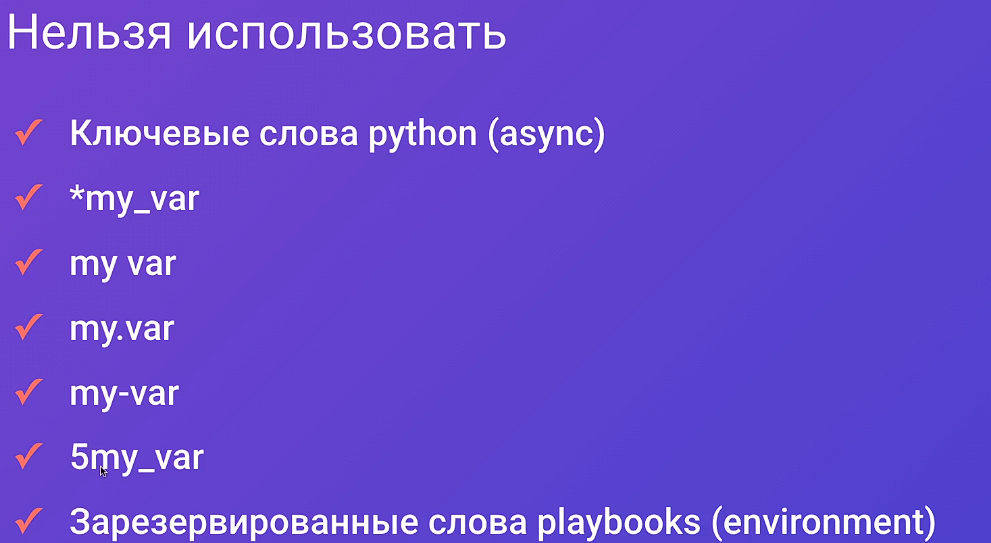
//пользователь существует, изменений не произошло;

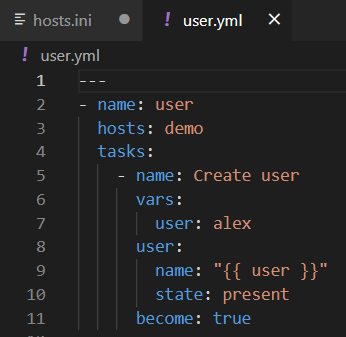


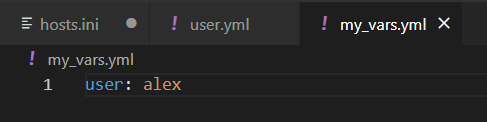
//добавляем в файл user.yml Become: true и запускаем с sudo – правами;

# Переменные

****

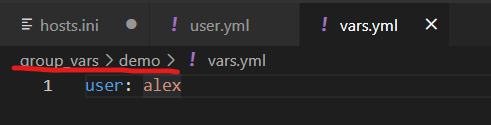
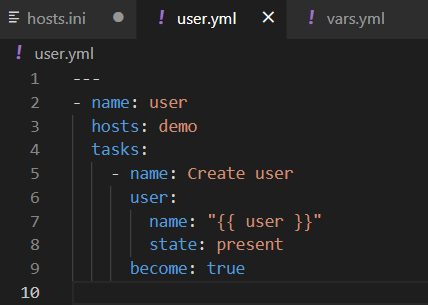
****

 //Задаём переменную **user: alex** ;

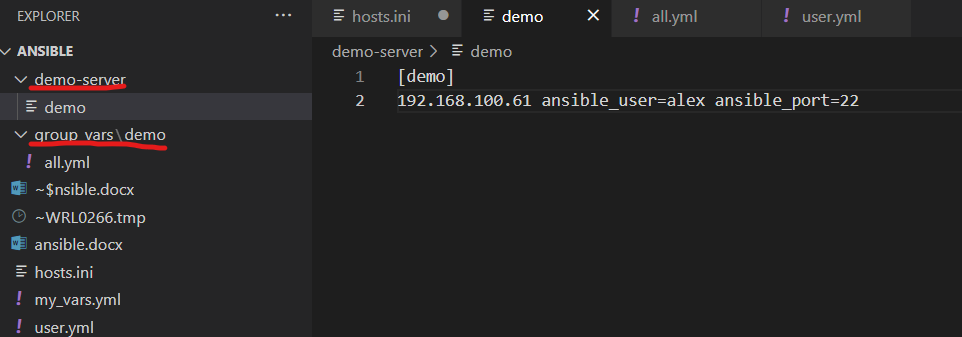


//задаем переменные с помощью файла **my\_vars.yml ;**

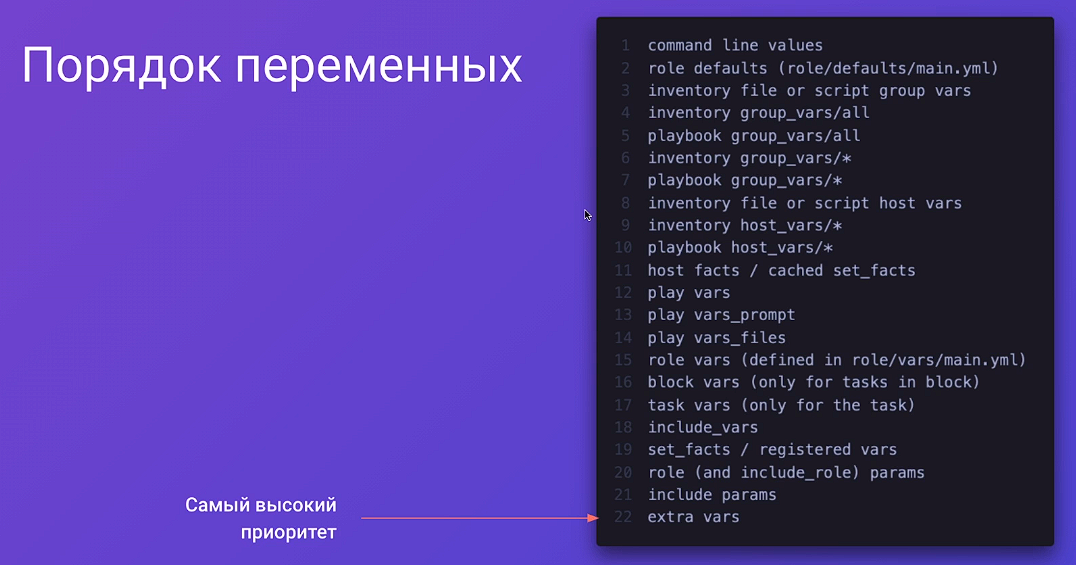
* **ansible-playbook -i hosts.ini user.yml –K // запускаем playbook ;**



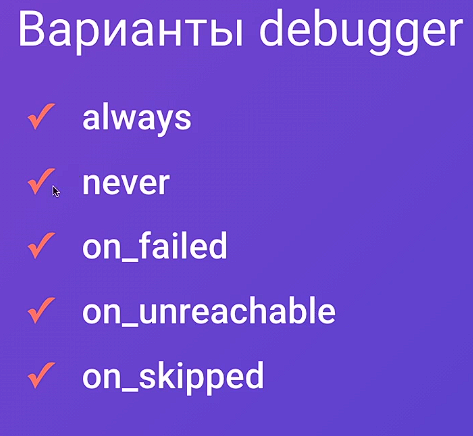
//задаём переменные с помощью плагина: создается папка **group\_vars** , в ней создается папка **demo** (название нашего инвентаря) и там создается файл **vars.yml** с нашими переменными;



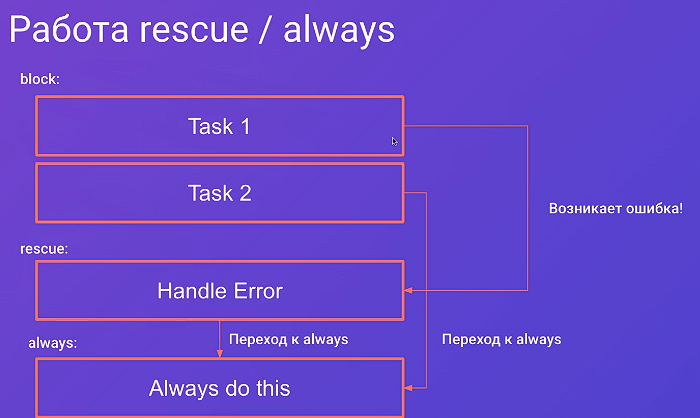
//группировка серверов и переменных в отдельные папки;



# Отладка



**Блоки и обработка ошибок**

****

---

- name: user

  hosts: demo

  tasks:

    - name: Preconfig block //создаем блок и добавляем в него таски

      block:

        - name: Create user //таск1 – создаем пользователя;

          user: alex2

            name: "{{ user }}"

            state: present

          register: error //записываем ошибки

//ignore\_error: yes //игнорим ошибку, если появится

        - name: Install curl //таск2 – установка curl;

          apt:

            name: curl

            update-cache: yes

          register: error

      become: true //запускается с требованием пароля админа;

when: ansible\_facts['distribution'] == 'Ubuntu' //таски выполнятся при условии дистрибутива Ubuntu;

      rescue:

        - name: Some error print

          debug:

            var: error //выполняется в случае ошибки;

      always:

        - name: Reboot // перезагрузка в любом случае;

          debug:

            msg: "Reboooooting" //выводит сообщение;

# Асинхронные задачи

****

---

- name: user

  hosts: demo

  tasks:

    - name: Preconfig block

      block:

       - name: Sleep

         command: /bin/sleep 15

         async: 1000  //кол-во секунд выполнения запроса

         poll:5       //каждые 5 секунд будет проверяться

---

- name: user

  hosts: demo

  tasks:

    - name: Preconfig block

      block:

       - name: Sleep

         command: /bin/sleep 15

         async: 1000  //кол-во секунд выполнения запроса

         poll:0       //означает, что мы выполняем следующий таск, не ожидая завершения предыдущего;

         register: sleep

      - debug:

          var: sleep

       - name: Echo

         command: echo "DONE"

      become: true

    - name: Check sleep status

      async\_status:

        jid: "{{ sleep.ansible\_job\_id }}" //проверяем статус нашего jobid в таске Sleep;

      register: job\_result

      until: job\_result.finished //ждём пока задача не будет finished;

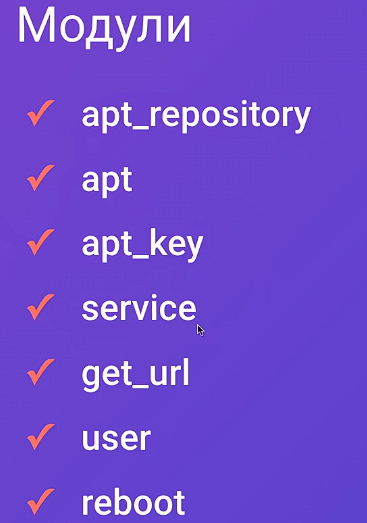
      retries: 100  //выполняем проверку максимум 100 раз  с разницей 1 сек.

      delay: 1

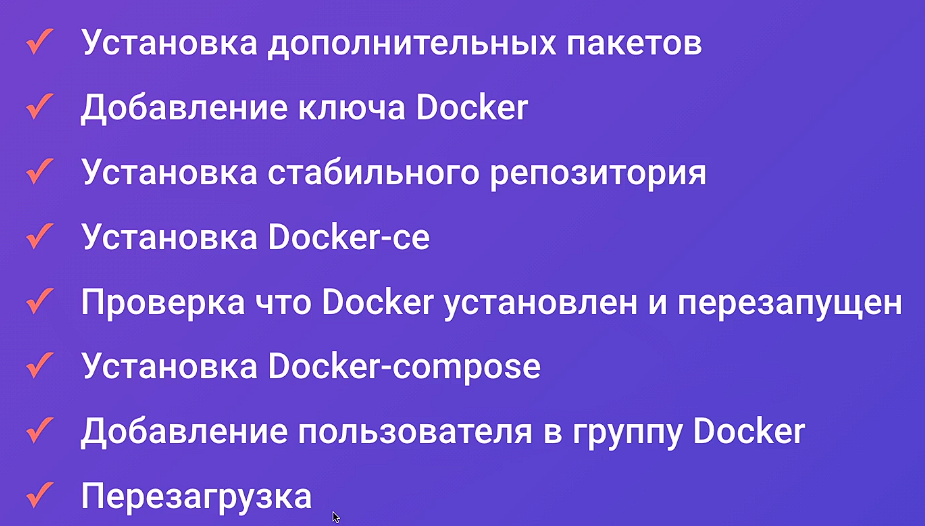
      become: true

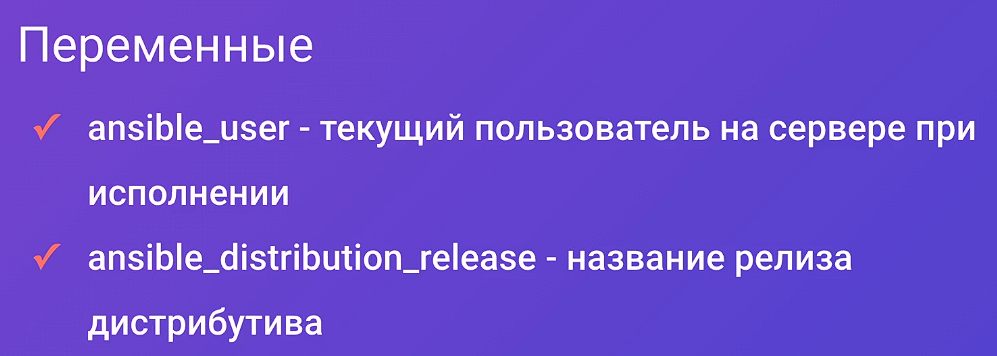
* **ansible-playbook -i demo-server user.yml –K // запускаем playbook с хостами в файле demo-server;**

# Упражнение: пишем настройку сервера:

****

**Задачи:**

****

****

---

- name: Preconfig

  hosts: demo

  tasks:

    - name: Docker install

      block:

        - name: Adding universe

          apt\_repository:

            repo: "deb http://us.archive.ubuntu.com/ubuntu/ {{ ansible\_distribution\_release }} universe" //добавляем нужный репозиторий;

            state: present

        - name: Install additional packets //установка доп.пакетов

          apt:

            name:

              - apt-transport-https

              - ca-certificates

              - curl

              - gnupg

              - lsb-release

            update-cache: yes

            cache\_valid\_time: 86400

        - name: Adding docker key

          apt\_key:

            url: https://download.docker.com/linux/ubuntu/gpg

            state: present

        - name: Add stable repository

          apt\_repository:

            repo: "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu {{ ansible\_distribution\_release }} stable"

            state: present

            update-cache: yes

            filename: docker

        - name: Install docker-ce //установка докера;

          apt:

            name: docker-ce

            update-cache: yes

        - name: Check if docker installed and rebooted //проверяем службу докера;

          service:

            name: docker

            state: restarted

            enabled: yes

      become: yes

    - name: Docker-compose install

      block:

        - name: Get the latest version on Docker Compose

          uri:

            url: https://api.github.com/repos/docker/compose/releases/latest

            body\_format: json

          register: page

        - name: Install docker Compose

          get\_url:

            url: "https://github.com/docker/compose/releases/download/{{ page.json.tag\_name }}/docker-compose-linux-x86\_64"

            dest: /usr/local/bin/docker-compose

            mode: 0755 //добавляем права на выполнение этого файла;

      become: yes

    - name: Installation finishing

      block:

        - name: Adding user in group Docker //доб. Пользователя в группу Докер;

          user:

            name: "{{ansible\_user}}"

            groups: docker

            append: yes

        - name: Server Reboot

          reboot:

      become: yes

* **ansible-playbook -i demo-server user.yml –K // запускаем playbook с хостами в файле demo-server;**

# Ansible Lint – это утилита для проверки корректности синтаксиса плейбука и стиля кода, которую можно интегрировать в CI-сервис.

* pip3 install ansible-lint

# Роли

