Репозиторий с Ansible

CI/CD на основе GitLab

REC **Проверить, идет ли запись**

Меня хорошо видно & слышно?

Кузнецов Алексей

cloud engineer

- 14 лет в IT
- 9 лет devops инженер
- создавал и поддерживал инфраструктуру для приложений в 3х крупный компаниях
- делаю вклад в open source
- выступаю на конференциях

@windblow

koyotne@bk.com

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в учебной группе #**OTUS CICD-GitLab-2025-04**

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Карта курса

СТАРТ

CI/CD - системы, подходы и workflow

Gitlab CI

Безопасность

Проектная работа

ФИНИШ

Маршрут вебинара

- Введение в Ansible и роль репозиториев
- Структура и организация репозитория Ansible
- Интеграция Ansible с GitLab CI
- Варианты реализации проектов на Ansible
- Практика
- Рефлексия

Цели вебинара

К концу занятия вы сможете

1. Освоить создание структурированного репозитория для запуска Ansible playbook

2. Изучить интеграцию Ansible с GitLab CI для автоматизации развертывания

3. Познакомиться с лучшими практиками организации и реализации проектов на Ansible

Кто из вас уже знаком с ansible и как практики автоматизации используете?

Введение в Ansible и роль репозиториев

Введение в Ansible и роль репозиториев

Ansible — это open-source инструмент для автоматизации IT-процессов, таких как управление конфигурациями, развертывание приложений и оркестрация сложных систем.

Принцип работы:

1. Использует декларативный подход.
2. Работает через SSH.
3. Основной язык: YAML.

Преимущества:

1. Простота.
2. Гибкость.
3. Идемпотентность.

Введение в Ansible и роль репозиториев

Роль репозитория:

1. Централизованное хранение всех компонентов Ansible.
2. Обеспечение версионности через системы контроля версий (например, Git).
3. Упрощение совместной работы команды

Проблемы без репозитория:

1. Хаотичное хранение скриптов и конфигураций.
2. Сложности с воспроизводимостью и масштабированием.
3. Риск потери данных или конфликтов при ручном управлении.

Введение в Ansible и роль репозиториев

Компоненты Ansible в репозитории:

- Playbooks
- Роли
- Инвентаризация
- Переменные
- Файлы и шаблоны

Базовый пример

Инициализация репозитория:

git init ansible-repo

cd ansible-repo

mkdir playbooks inventory

touch playbooks/setup_server.yml

touch inventory/hosts

git add .

git commit -m "Initial Ansible repo setup"

Базовый пример

Простой playbook:

```
name: Setup basic server # название playbook, которое описывает его цель.
hosts: all # указывает, на каких хостах (серверах) будет выполняться playbook.
become: yes # запуск модуля с повышенными привилегиями (по-умолчанию root)
tasks: # определяет список задач, которые Ansible должен выполнить.
  - name: Install basic packages # название конкретной задачи внутри playbook.
    apt: # вызывает модуль Ansible apt для управления пакетами на системах, использующих пакетный менеджер APT (Debian, Ubuntu и т. д.).
      name: "{{ item }}" # указывает имя пакета для установки.
      state: present # указывает желаемое состояние пакета.
    loop: # определяет цикл, который позволяет выполнить задачу apt для каждого элемента
      - vim
      - htop
```

Вопросы

+ если есть вопросы

– если вопросов нет

Введение в Ansible и роль репозиториев

Структура и организация репозитория Ansible

Репозиторий Ansible — это каталог, содержащий все необходимые файлы для автоматизации: playbooks, роли, переменные, инвентаризацию и т. д.

```
ansible-repo/
├── playbooks/
├── roles/
├── inventory/
├── group_vars/
├── host_vars/
├── files/
├── templates/
├── ansible.cfg
└── requirements.yml
```

Структура и организация репозитория Ansible

playbooks/

Хранит основные YAML-файлы (playbooks), которые описывают сценарии автоматизации.

Файлы с расширением .yml, где определены хосты, роли и задачи.

```
- name: Deploy web application
  hosts: webservers
  become: yes
  roles:
    - webserver
  tasks:
    - name: Ensure nginx is running
      service:
        name: nginx
        state: started
```

Структура и организация репозитория Ansible

roles/

Содержит модульные, повторно используемые блоки кода — роли.

```
roles/
├── webserver/
│   ├── tasks/
│   │   └── main.yml
│   ├── templates/
│   │   └── nginx.conf.j2
│   ├── files/
│   │   └── app.conf
│   ├── defaults/
│   │   └── main.yml
│   ├── handlers/
│   │   └── main.yml
│   └── vars/
│       └── main.yml
```

Структура и организация репозитория Ansible

roles/

Подкаталоги:

- tasks/: Основные задачи (например, установка пакетов, запуск сервисов).
- templates/: Шаблоны Jinja2 для динамических конфигураций.
- files/: Статические файлы для копирования на хосты.
- defaults/: Переменные по умолчанию (с низким приоритетом).
- handlers/: Обработчики, запускаемые по событиям (например, перезапуск сервиса).
- vars/: Переменные с высоким приоритетом.

Роли можно переиспользовать в разных playbooks или проектах.

Структура и организация репозитория Ansible

inventory/

Хранит списки хостов и групп для управления.

Типы:

- Статический: Текстовый файл (INI или YAML) с IP-адресами или именами хостов.
- Динамический: Скрипты или плагины для генерации хостов (например, из YandexCloud, Cloud.ru).

```
[webservers]
web1.example.com
web2.example.com
[dbservers]
db1.example.com
[all:vars]
ansible_user=admin
ansible_ssh_private_key_file=~/.ssh/id_rsa
```

Структура и организация репозитория Ansible

group_vars/

Хранение переменных для групп хостов или отдельных машин.

Переменные для групп из инвентаризации (например, webservers).

Пример: *group_vars/webservers.yml*

```
nginx_port: 80
app_version: ‘2.1.3’
```

host_vars/

Переменные для конкретных хостов.

Пример: *host_vars/web1.example.com.yml*

```
server_name: web1.example.com
max_connections: 1000
```

Структура и организация репозитория Ansible

files/

Хранит статические файлы, которые Ansible копирует на целевые хосты.

Пример: копирование конфигурации: files/app.conf переносится на сервер.

```
- name: Copy app config
  copy:
    src: files/app.conf
    dest: /etc/app/app.conf
```

Структура и организация репозитория Ansible

templates/

Хранит шаблоны Jinja2 для динамической генерации файлов.

Использует переменные для кастомизации.

Пример: templates/nginx.conf.j2

```
server {
    listen {{ nginx_port }};
    server_name {{ server_name }};
    location / {
        root /var/www/html;
    }
}
```

Структура и организация репозитория Ansible

templates/

Задача в playbook

```
- name: Render nginx config
  template:
    src: templates/nginx.conf.j2
    dest: /etc/nginx/nginx.conf
  notify: Restart nginx
```

Структура и организация репозитория Ansible

ansible.cfg

Файл конфигурации Ansible для настройки поведения.

Упрощает запуск, задает пути и параметры по умолчанию.

```
[defaults]
inventory = ./inventory
roles_path = ./roles
remote_user = admin
host_key_checking = False
```

Структура и организация репозитория Ansible

requirements.yml

Список внешних ролей или коллекций с Ansible Galaxy.

Подключает готовые роли для ускорения работы.

```
- src: geerlingguy.nginx
  version: 3.1.2
```

Команда запуска

ansible-galaxy install -r requirements.yml

Структура и организация репозитория Ansible

- **Модульность**: Используйте роли для разделения задач.
- **Именование**: Называйте файлы и каталоги понятно (например, deploy_app.yml, webserver).
- **Версионность**: Инициализируйте Git-репозиторий.
- **Разделение сред**: Храните production и staging в отдельных файлах инвентаризации.
- **Безопасность**: Не храните пароли и ключи в репозитории, используйте Ansible Vault для шифрования.

Вопросы

+ если есть вопросы

– если вопросов нет

Интеграция Ansible с GitLab CI

Интеграция Ansible с GitLab CI

GitLab CI — это встроенный инструмент непрерывной интеграции и доставки (CI/CD) в GitLab.

Зачем интегрировать Ansible с GitLab CI?

Основы работы GitLab CI

1. **Пайплайн**
2. **Ключевые элементы**
3. **Процесс**

Интеграция Ansible с GitLab CI

Подготовка к интеграции

Требования:

- Репозиторий Ansible с playbook’ами, ролями, инвентаризацией (см. структуру репозитория).
- Доступ к GitLab (self-hosted или gitlab.com).
- Настроенный GitLab Runner с поддержкой Docker или shell для выполнения заданий.

Установка Ansible в pipeline:

- Используйте Docker-образ с Python и Ansible (например, python:3.9).
- Установите Ansible через pip в задании.

Безопасность:

- Храните чувствительные данные (пароли, SSH-ключи) в переменных GitLab CI (Settings > CI/CD > Variables).
- Используйте Ansible Vault для шифрования секретных переменных.

Интеграция Ansible с GitLab CI

Пример файла .gitlab-ci.yml

```
stages:
   - test
   - deploy
test_playbook:
   stage: test
   image: python:3.9
   before_script:
      - pip install ansible
      - pip install ansible-lint
   script:
      - ansible-lint playbooks/*.yml
      - ansible-playbook -i inventory/staging playbooks/deploy_app.yml --syntax-check
   only:
       - main
       - develop
```

Интеграция Ansible с GitLab CI

```
deploy_staging:
   stage: deploy
   image: python:3.9
   before_script:
      - pip install ansible
   script:
      - ansible-playbook -i inventory/staging playbooks/deploy_app.yml
   environment:
      name: staging
   only:
      - develop
deploy_production:
   stage: deploy
   image: python:3.9
   before_script:
      - pip install ansible
   script:
      - ansible-playbook -i inventory/production playbooks/deploy_app.yml
   environment:
      name: production
   when: manual
   only: - main
```

Интеграция Ansible с GitLab CI

Настройка окружения

Инвентаризация

Храните файлы в inventory/, например, inventory/staging и inventory/production.

```
# inventory/staging
[webservers]
staging1.example.com
```

Переменные

В GitLab: Добавьте SSH-ключ (SSH_PRIVATE_KEY) и пользователя (ANSIBLE_USER) в CI/CD Variables. Используйте в playbook’ах через group_vars/ или host_vars/.

Интеграция Ansible с GitLab CI

Ansible Vault

Зашифруйте секреты

ansible-vault encrypt group_vars/secrets.yml

Храните пароль Vault в защищенной переменной GitLab CI (например, VAULT_PASSWORD).
Используйте в пайплайне

```
script:
  - echo "$VAULT_PASSWORD" > vault_pass.txt
  - ansible-playbook -i inventory/production playbooks/deploy_app.yml
--vault-password-file vault_pass.txt
```

Интеграция Ansible с GitLab CI

Процесс работы

1. **Коммит**.
2. **Тестирование**.
3. **Развертывание**
4. **Мониторинг**.

Вопросы

+ если есть вопросы

– если вопросов нет

Варианты реализации проектов на Ansible

Варианты реализации проектов на Ansible

Ansible проекты могут быть реализованы разными способами в зависимости от масштаба, сложности и требований.

1. Монолитный подход:
Все задачи, переменные и конфигурации собраны в одном или нескольких playbook’ах.
Минимальная структура: один YAML-файл с задачами и инвентаризация.

Плюсы:
- Простота
- Быстрая разработка
- Подходит для разовых задач (например, настройка одного сервера).

Минусы:
- Сложно масштабировать
- Трудно поддерживать
- Нет повторного использования.

Когда использовать?

Варианты реализации проектов на Ansible

Пример

```
- name: Setup a simple server
  hosts: all
  become: yes
  vars:
    app_version: "1.2.3"
    nginx_port: 80
  tasks:
    - name: Install nginx
      apt:
        name: nginx
        state: present
    - name: Copy config
      copy:
        content: |
          server {
            listen {{ nginx_port }};
            location / {
              root /var/www/html;
            }
          }
        dest: /etc/nginx/nginx.conf
    - name: Start nginx
      service:
        name: nginx
        state: started
```

Варианты реализации проектов на Ansible

2. Модульный подход с ролями

Задачи, переменные, файлы и шаблоны разделены на роли.

Роли — независимые модули, которые можно переиспользовать в разных playbooks.

```
ansible-repo/
├── playbooks/
│   └── deploy_app.yml
├── roles/
│   ├── webserver/
│   │   ├── tasks/
│   │   │   └── main.yml
│   │   ├── templates/
│   │   │   └── nginx.conf.j2
│   │   └── defaults/
│   │       └── main.yml
│   ├── database/
│   │   ├── tasks/
│   │   │   └── main.yml
│   │   └── defaults/
│   │       └── main.yml
├── inventory/
└── group_vars/
```

Варианты реализации проектов на Ansible

Пример роли

roles/webserver/tasks/main.yml

```
- name: Install nginx
  apt:
    name: nginx
    state: present
- name: Render nginx config
  template:
    src: nginx.conf.j2
    dest: /etc/nginx/nginx.conf
  notify: Restart nginx
```

Варианты реализации проектов на Ansible

Плюсы:
- Модульность
- Читаемость
- Легкая поддержка

Минусы:
- Требует больше времени на начальную настройку.
- Сложнее для новичков.

Когда использовать?

Варианты реализации проектов на Ansible

Лучшие практики

Выбор подхода:
Монолитный: для простых, разовых задач.
Модульный: для масштабируемых, командных проектов.

Стандарты:
Используйте ansible-lint для проверки стиля и ошибок.
Задокументируйте роли и playbooks.

Интеграция:
Подключайте к CI/CD (например, GitLab CI) для тестирования и развертывания.
Храните репозиторий в Git для версионности.

Безопасность:
Используйте Ansible Vault для шифрования секретных данных.
Ограничивайте доступ к inventory и переменным.

Вопросы

+ если есть вопросы

– если вопросов нет

LIVE

Пример использования

Практика

Описание этапов:

1. Инициализировать Git-репозиторий и создать структуру для Ansible.
2. Написать playbook для установки Nginx.
3. Настроить инвентаризацию для тестовой среды.
4. Создать pipeline GitLab CI для проверки и развертывания.
5. Протестировать CD-процесс.

Ключевые тезисы занятия

1. **Структурирование репозитория Ansible**: Освойте создание организованного репозитория с playbook’ами, ролями, инвентаризацией и переменными для эффективной автоматизации и масштабируемости.
2. **Автоматизация с GitLab CI**: Настройте Continuous Deployment (CD) с помощью GitLab CI, чтобы автоматически проверять и применять Ansible playbooks для надежного развертывания инфраструктуры.

Вопросы

+ если есть вопросы

– если вопросов нет

Подведём итоги занятия

Теперь вы сможете:

Отправьте в чат номера достигнутых целей

1. Освоить создание структурированного репозитория для запуска Ansible playbooks

2. Изучить интеграцию Ansible с GitLab CI для автоматизации развертывания

3. Познакомиться с лучшими практиками организации и реализации проектов на Ansible

Вопросы для проверки

1. Зачем нужен репозиторий для работы с Ansible?
2. В чем разница между group_vars и host_vars? Какие переменные приоритетнее?
3. Что означает параметр become: yes в playbook?

Отправьте в чат эмодзи, который отражает ваше настроение после занятия

Продолжите высказывание: теперь я умею/знаю...

Что планируете использовать в работе?

Заполните, пожалуйста, опрос о занятии

Мы читаем все ваши сообщения
и берем их в работу ✏️❤️

СКЛАДЧИК

ЭТОТ КОНТЕНТ КУПЛЕН НА САЙТЕ SKLADCHIK.ORG

SKL.TO

ТЫ ЕЩЕ НЕ В КЛУБЕ?
 присоединяйся и начинай экономить!

ЧТО ТАКОЕ КЛУБ «СКЛАДЧИК»?

Платформа
Это платформа, где каждый день тысячи людей собираются вместе, чтобы общими усилиями находить, приобретать и изучать курсы интересные именно им.

Сообщество
Это сообщество из 500 000 людей, открывших для себя выгодный способ быть в тренде самых актуальных и получать ценные знания по минимальной цене.

Библиотека
Это крупнейшая библиотека инфопродуктов в которой можно найти практически любой курс или тренинг продававшийся за последние 10 лет, а также уникальные авторские инфопродукты, которые не получиться найти больше нигде.