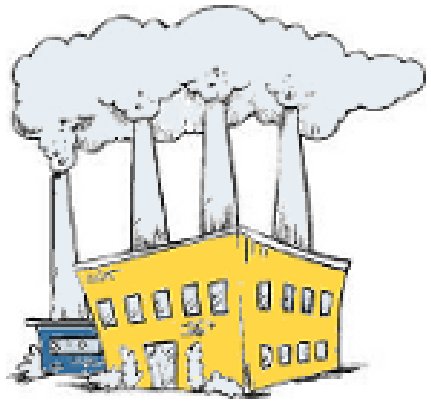
Fabric - Автоматизируем деплой

DevOps для ленивых



fabfile.py → >fab deploy

```python
# coding: utf-8
import os
from fabric.api import run, env, cd, roles

# Списком можно перечислить несколько серверов, которые у вас считаются "продакшеном"
env.roledefs['production'] = ['git@example.org:2244']

def production_env():
    """Окружение для продакшена"""
    env.key_filename = [os.path.join(os.environ['HOME'], '.ssh', 'git_example_org')]  # Локальный
 путь до файла с ключами
    env.user = 'git'  # На сервере будем работать из под пользователя "git"
    env.project_root = '/home/username/work/project'  # Путь до каталога проекта (на сервере)
    env.shell = '/usr/local/bin/bash -c'  # Используем шелл отличный от умолчательного (на сервер
е)
    env.python = '/home/username/work/project/venv/bin/python'  # Путь до python (на сервере)


@roles('production')
def deploy():
    production_env()  # Инициализация окружения
    with cd(env.project_root):  # Заходим в директорию с проектом на сервере
        run('git pull origin master')  # Пуляемся из репозитория
        run('find . -name "*.mo" -print -delete')  # Чистим старые скомпиленные файлы gettext'а
        run('{} manage.py compilemessages'.format(env.python))  # Собираем новые файлы gettext'а
        run('{} manage.py collectstatic --noinput'.format(env.python))  # Собираем статику

@roles('production')
def pip_install():
    production_env()
    run('{pip} install --upgrade -r {filepath}'.format(pip=env.pip,
        filepath=os.path.join(env.project_root, 'requirements.txt')))
```

+ работал через 5 минут после того, как первый раз его прикрутил, низкий порог входа

+ каждый шаг удобно и понятно логируется

- Не нашел в коробке прямого деплоя на сервер, только через github. Не автоматизирует, если код в Bitbucket Sigma :)