

Пишем пайплайны

- [Пишем](#)
- [Плагины](#)

Пишем

Пример пайплайна:

```
kind: pipeline
type: docker
name: default

steps:
- name: greeting
  image: alpine
  commands:
  - echo hello
  - echo world
```

Можно указывать в одном drone.yml'е несколько пайплайнов/секретов/и т.д.

- kind: определяет что это - пайплайн, секрет или еще что
- type: тип пайплайна (exec, docker, etc...)
- name: имя пайплайна
- steps: шаги пайплайна

Приколюхи

Можно фильтровать прод и тестинг таким образом:

```
node:
  target: shell-scripts
when:
  branch:
    - master
```

В node указывается тег target, которым отделять прод от теста. Например на проде:

```
DRONE_RUNNER_LABELS=target:prod
```

А на тестинге:

```
DRONE_RUNNER_LABELS=target:testing
```

Потом сделать 2 пайплайна для разных веток (testing/master). Правда у меня почему-то не вышло.

Как это все строить?

Деплой по сути состоит из нескольких этапов - загрузка кода, проверка+-линтовка, деплой (опционально с доп. подтверждением).

Плагины

Указываются как image:

```
kind: pipeline
type: docker
name: default

steps:
- name: test
  image: golang:1.13
  commands:
  - go build
  - go test -v

- name: notify
  image: plugins/slack
  settings:
    channel: dev
    webhook: https://hooks.slack.com/services/...
```

Плагины

SCP (SSH Copy)

```
- name: scp files
image: appleboy/drone-scp
settings:
  host:
    from_secret: scp_host
  username:
    from_secret: scp_user
  password:
    from_secret: scp_pass
  port: 22
  source: /root/archlinux-*-x86_64.iso
  target: /var/www/mirror
```

Telegram notification

```
- name: send telegram notification
image: appleboy/drone-telegram
settings:
  token:
    from_secret: tg_token
  to:
    from_secret: tg_user
  message: >
    {{#success build.status}}
    build {{build.number}} of {{repo.name}} succeeded. Commit: {{commit.message}}. Duration: {{since
build.started}}. Good job.
    {{else}}
    build {{build.number}} of {{repo.name}} failed. Commit: {{commit.message}}. Duration: {{since
build.started}}. Fix me please.
    {{/success}}
```

when:

status:

- success

- failure