Portable Pipelines

Vilnius, DevOpsPro Europe 2019

Carlos León - @mongrelion

Strategic Consultant

All Things Cloud Native, DevOps, Programmable Infrastructure, Automation and Hang Gliding <3

Agenda

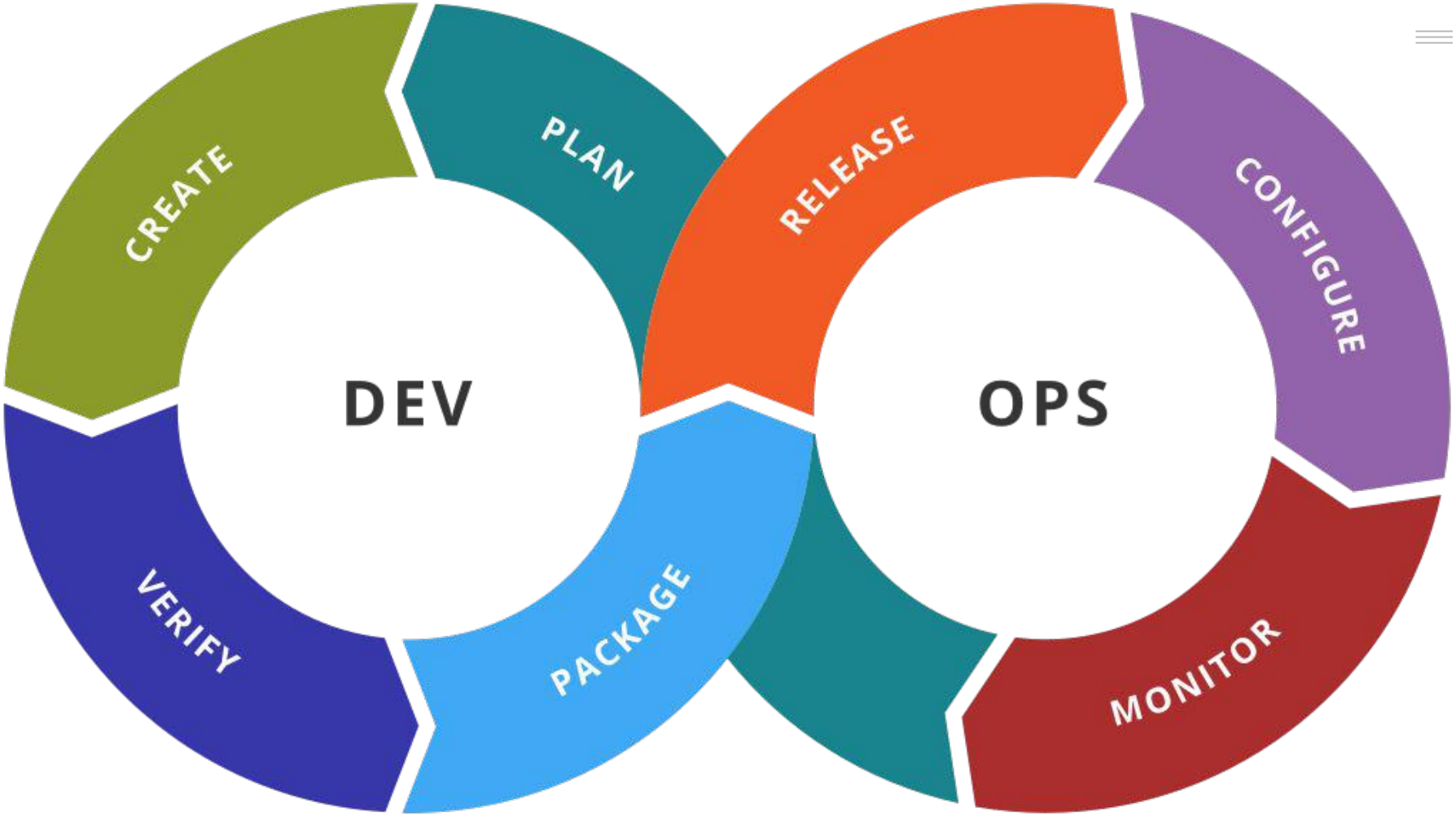
1. CICD: The Basics
2. The Status Quo
3. CICD Tooling
4. Example
5. Portability
6. Q&A

Questions

https://sli.do

#devops2019

CICD: The Basics



Build -> Test -> Deploy Staging -> More Tests -> Deploy Production

The Good

The Good

- Predictable

The Good

- Predictable
- Reproducible

The Good

- Predictable
- Reproducible
- Visibility

The Good

- Predictable
- Reproducible
- Visibility
- Accountability

The Good

- Predictable
- Reproducible
- Visibility
- Accountability
- Less error prone

The Bad

The Bad

- Semantic Versioning

The Bad

- Semantic Versioning
- Can get chaotic when the pipeline is slow

The Status Quo

The Good

The Good

- Community support

The Good

- Community support
- Commercial support

The Good

- Community support
- Commercial support
- Extensions/plugins ecosystem

The Good

- Community support
- Commercial support
- Extensions/plugins ecosystem
- Well known

The Good

- Community support
- Commercial support
- Extensions/plugins ecosystem
- Well known
- Battle tested

The Bad

The Bad

- Eager for resources

The Bad

- Eager for resources
- Hard to automate

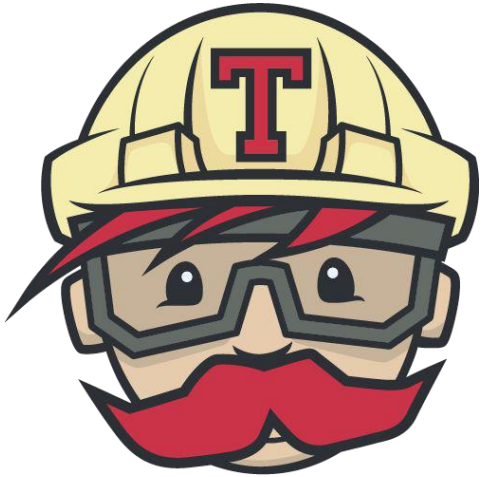
The Bad

- Eager for resources
- Hard to automate
- Stateful to the bone

The Bad

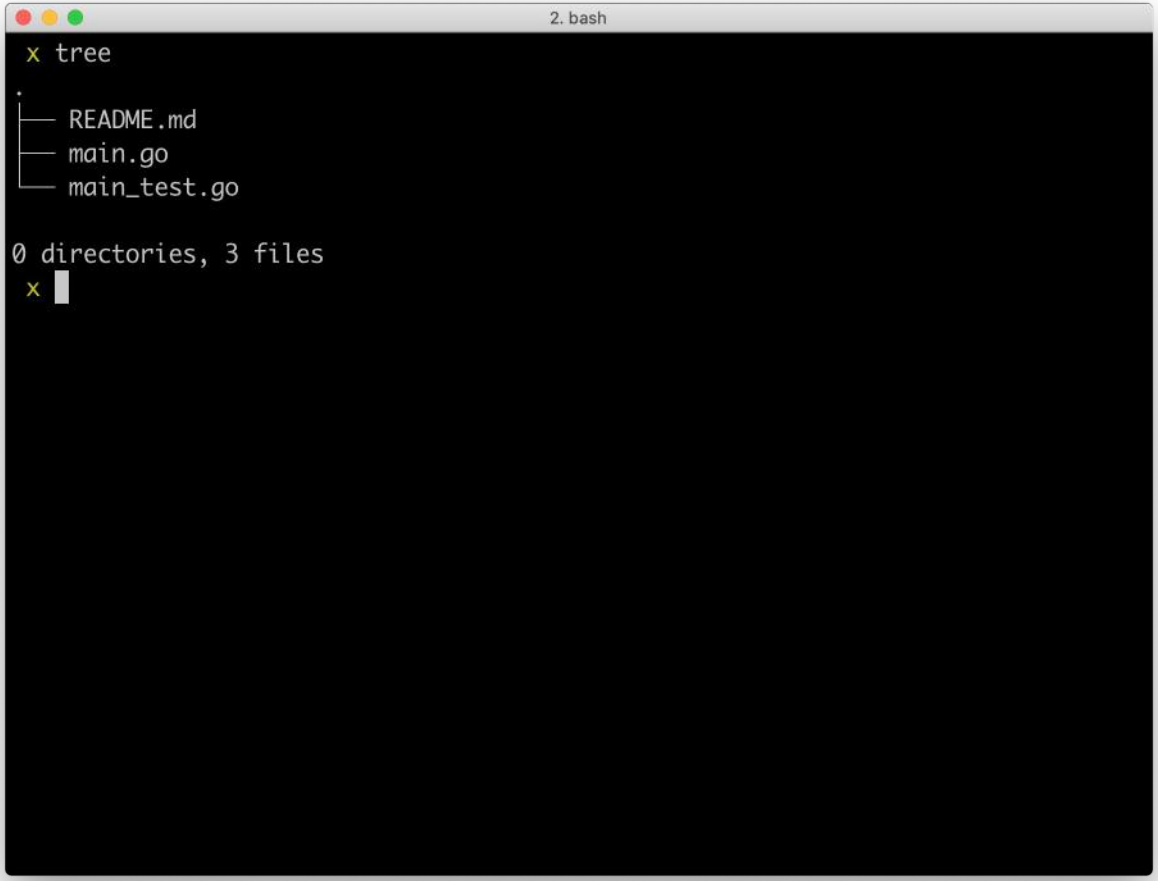
- Eager for resources
- Hard to automate
- Stateful to the bone
- Extensions/plugin ecosystem

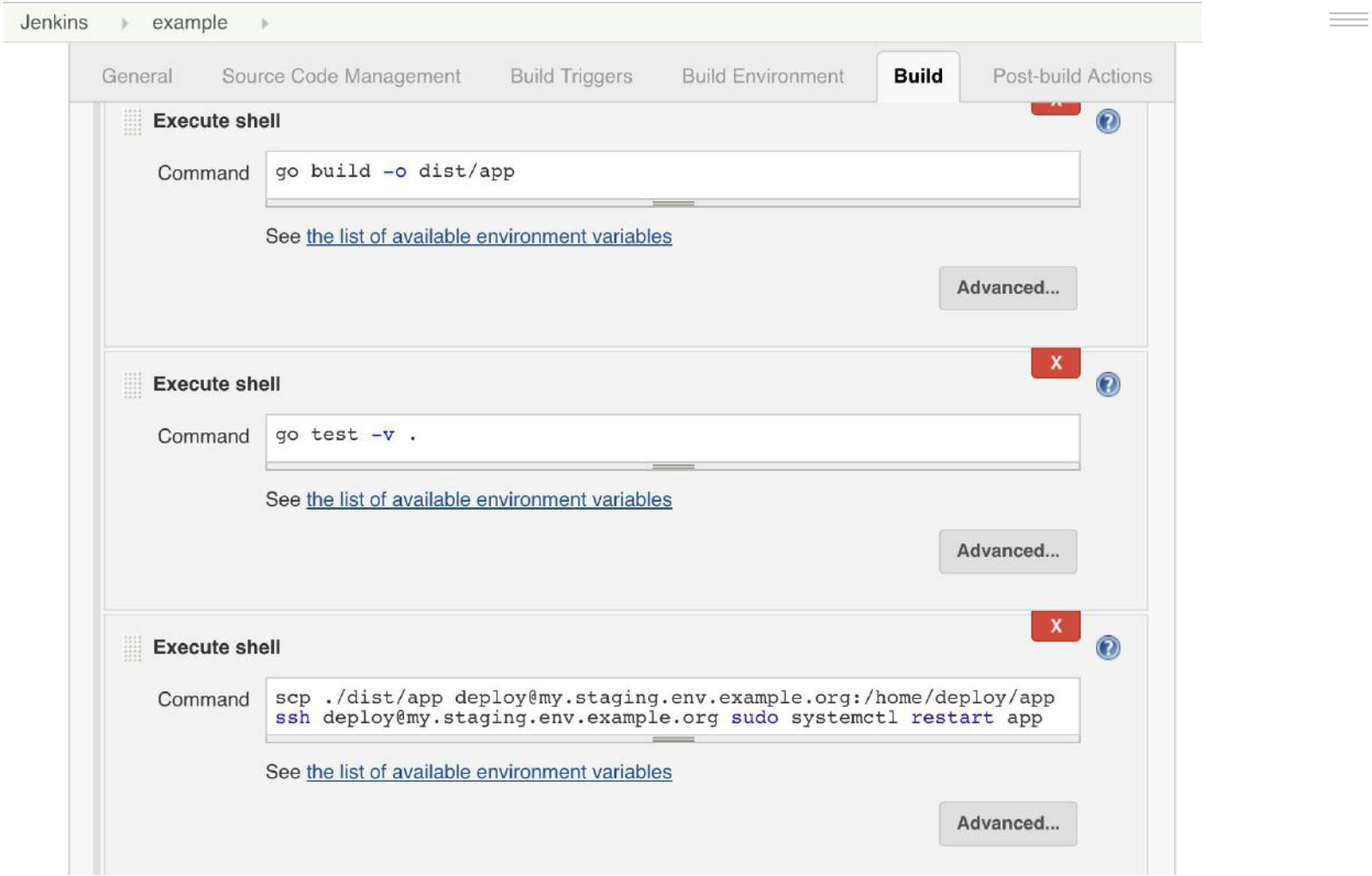
CICD Tooling



Example

Web application written in Go


2. bash
x tree
.
├── README.md
├── main.go
└── main_test.go
0 directories, 3 files
x


Jenkins
example
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Execute shell
Command
go build -o dist/app
See the list of available environment variables
Advanced...
Execute shell
Command
go test -v .
See the list of available environment variables
Advanced...
Execute shell
Command
scp ./dist/app deploy@my.staging.env.example.org:/home/deploy/app
ssh deploy@my.staging.env.example.org sudo systemctl restart app
See the list of available environment variables
Advanced...

Issues with this approach

Issues with this approach

- Lots of manual steps

Issues with this approach

- Lots of manual steps
- Changes are not visible

Issues with this approach

- Lots of manual steps
- Changes are not visible
- No track

Issues with this approach

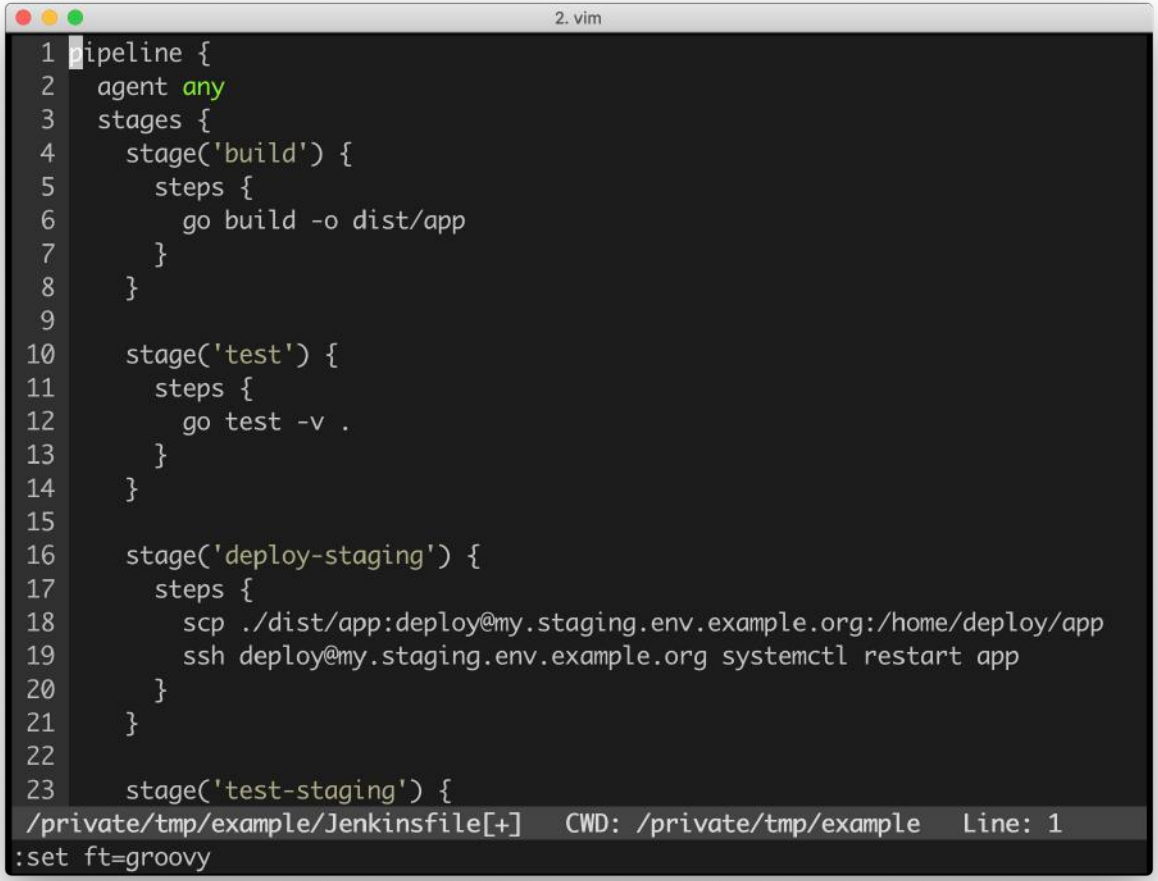
- Lots of manual steps
- Changes are not visible
- No track
- No responsibility

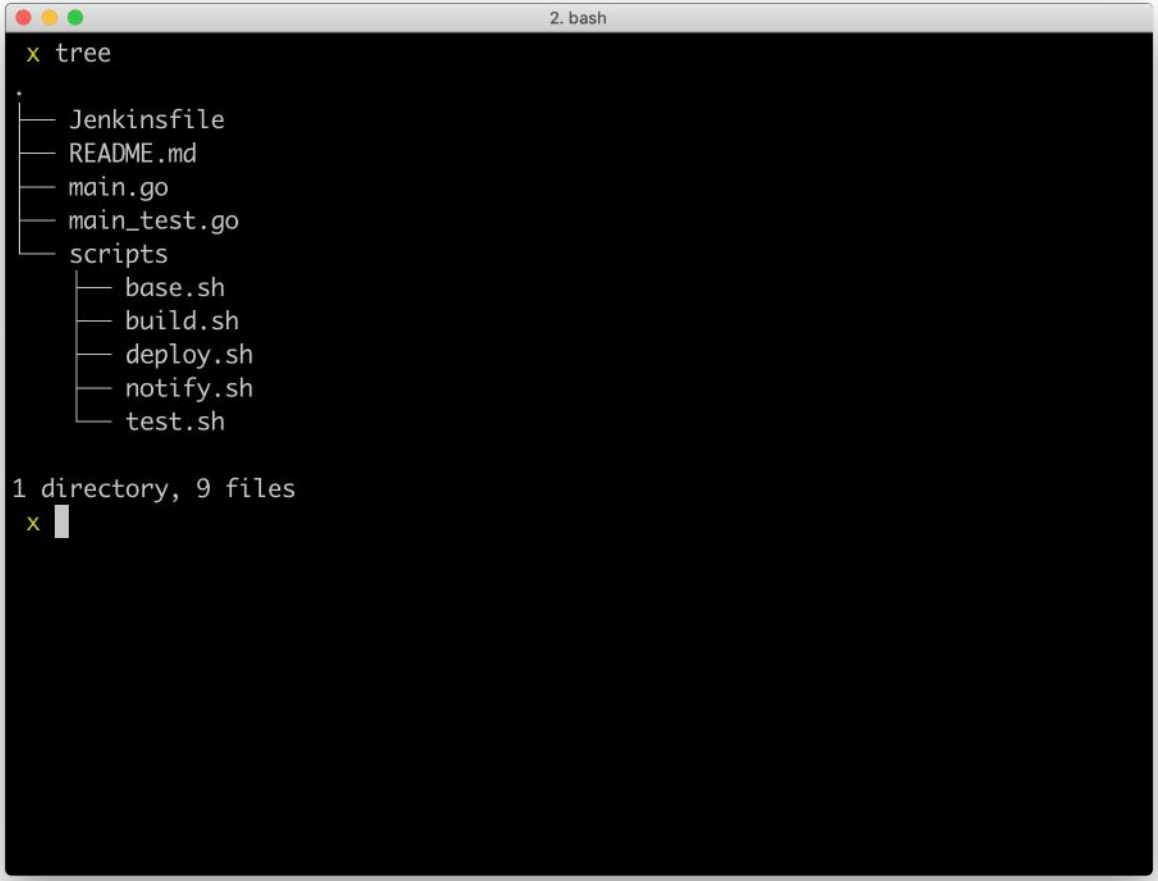
Issues with this approach

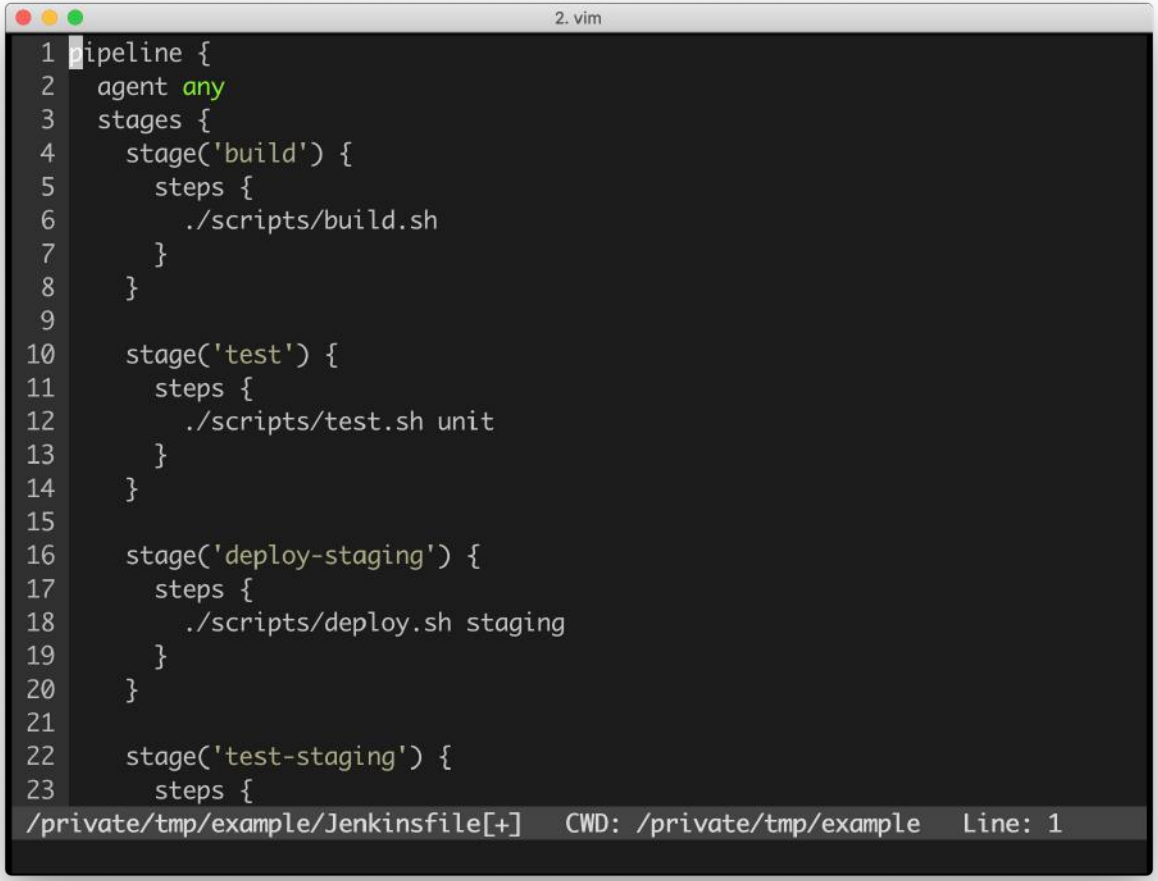
- Lots of manual steps
- Changes are not visible
- No track
- No responsibility
- Long feedback loops

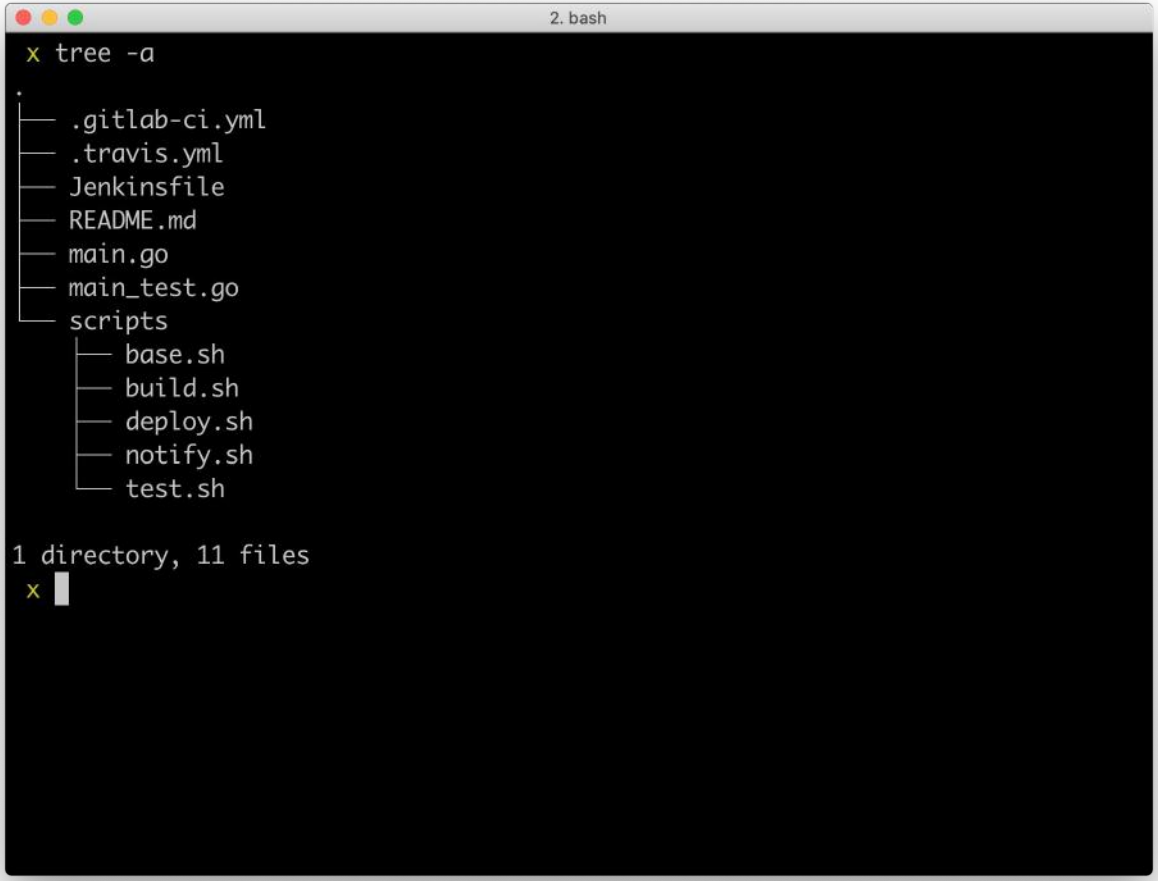
Example

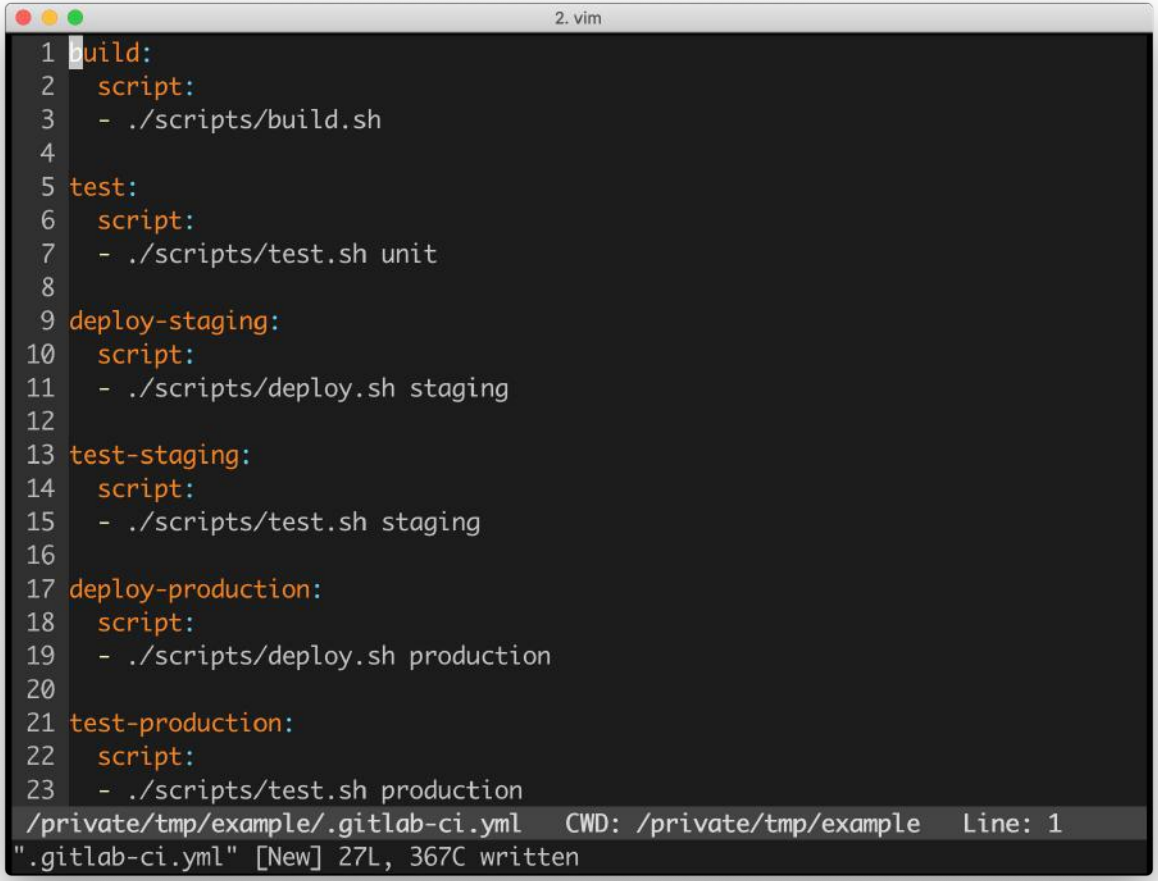
Portability

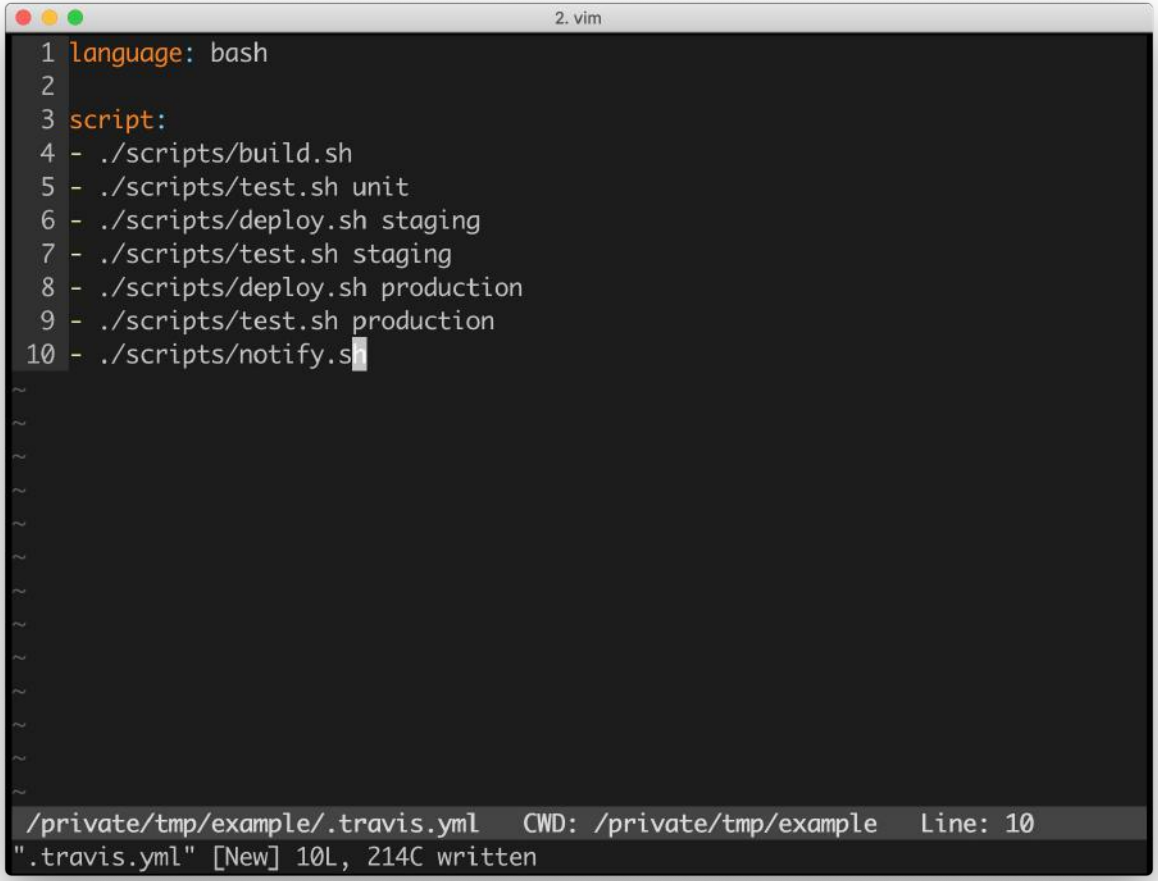

2. vim
pipeline {
agent any
stages {
stage('build') {
steps {
go build -o dist/app
}
}
stage('test') {
steps {
go test -v .
}
}
stage('deploy-staging') {
steps {
scp ./dist/app:deploy@my.staging.env.example.org:/home/deploy/app
ssh deploy@my.staging.env.example.org systemctl restart app
}
}
stage('test-staging') {
/private/tmp/example/Jenkinsfile[+] CWD: /private/tmp/example Line: 1
:set ft=groovy


2. bash
x tree
.
├── Jenkinsfile
├── README.md
├── main.go
├── main_test.go
└── scripts
    ├── base.sh
    ├── build.sh
    ├── deploy.sh
    ├── notify.sh
    └── test.sh
1 directory, 9 files
x


2. vim
pipeline {
agent any
stages {
stage('build') {
steps {
./scripts/build.sh
}
}
stage('test') {
steps {
./scripts/test.sh unit
}
}
stage('deploy-staging') {
steps {
./scripts/deploy.sh staging
}
}
stage('test-staging') {
steps {
/private/tmp/example/Jenkinsfile[+] CWD: /private/tmp/example Line: 1


2. bash
x tree -a
.
├── .gitlab-ci.yml
├── .travis.yml
├── Jenkinsfile
├── README.md
├── main.go
├── main_test.go
└── scripts
    ├── base.sh
    ├── build.sh
    ├── deploy.sh
    ├── notify.sh
    └── test.sh
1 directory, 11 files
x


2. vim
build:
script:
- ./scripts/build.sh
test:
script:
- ./scripts/test.sh unit
deploy-staging:
script:
- ./scripts/deploy.sh staging
test-staging:
script:
- ./scripts/test.sh staging
deploy-production:
script:
- ./scripts/deploy.sh production
test-production:
script:
- ./scripts/test.sh production
/private/tmp/example/.gitlab-ci.yml CWD: /private/tmp/example Line: 1
".gitlab-ci.yml" [New] 27L, 367C written


2. vim
1 language: bash
2
3 script:
4 - ./scripts/build.sh
5 - ./scripts/test.sh unit
6 - ./scripts/deploy.sh staging
7 - ./scripts/test.sh staging
8 - ./scripts/deploy.sh production
9 - ./scripts/test.sh production
10 - ./scripts/notify.sh
/private/tmp/example/.travis.yml CWD: /private/tmp/example Line: 10
".travis.yml" [New] 10L, 214C written

Pros

Pros

- Portability

Pros

- Portability
- Short iteration loops

Pros

- Portability
- Short iteration loops
- Accountability

Pros

- Portability
- Short iteration loops
- Accountability
- Trackable changes

Cons

Cons

- Hard to bootstrap

Cons

- Hard to bootstrap
- Time to learn bash

Cons

- Hard to bootstrap
- Time to learn bash
- Hard to reuse community components

Questions

https://sli.do

#devops2019

Thank you.