Gitlab CI + Docker

Ondrej Sika
ondrej@ondrejsika.com
@ondrejsika

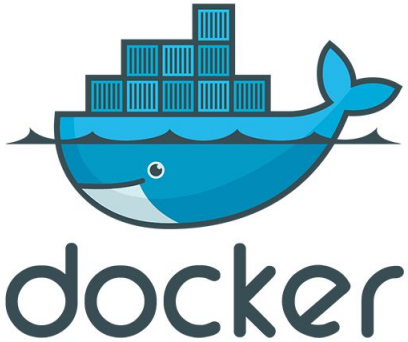
Linux Days 2018,
Prague, 6. 10. 2018

https://sika.link/linuxdays2018

Goals

- Build application
- Run tests
- Deploy to staging env.


træfik


docker

What is CI?

What is CI?

In software engineering, continuous integration is the practice of merging all developer working copies to a shared mainline several times a day.

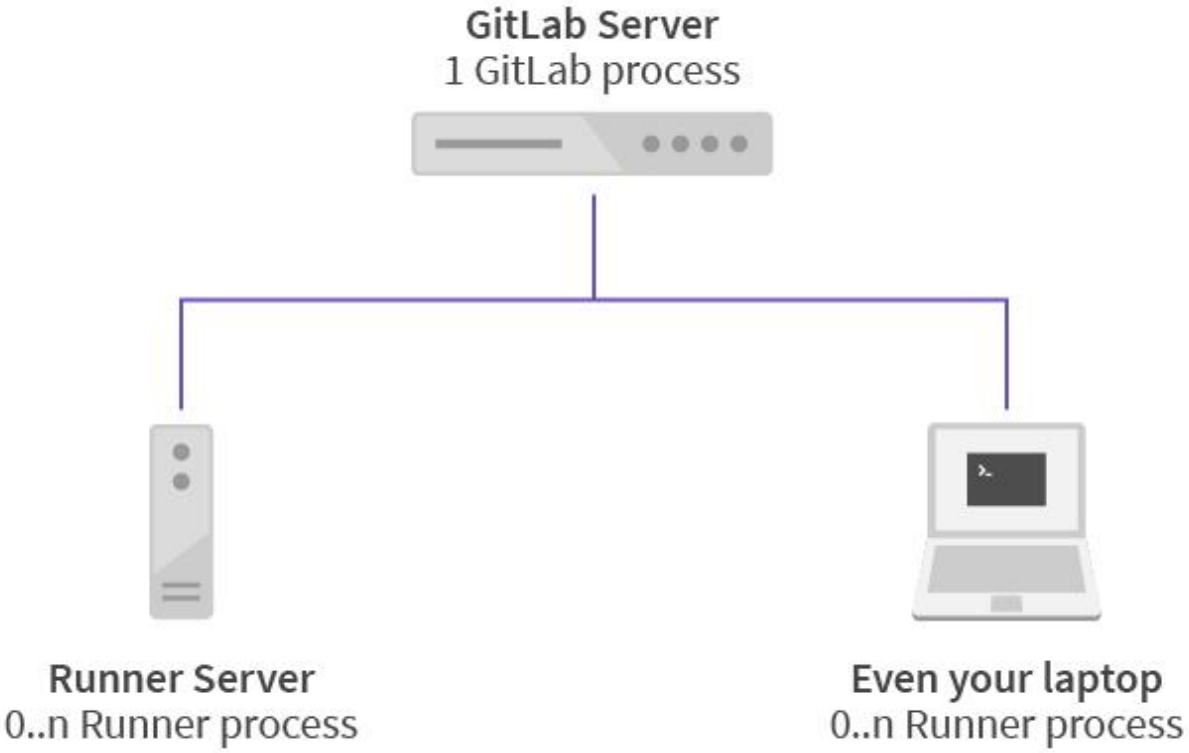
Usage of CI

Automatization of

- build process
- testing
- deployment
  - dev
  - staging
  - production
- code quality
  - Linting
  - Formating

Gitlab CI

Gitlab CI Architecture



Gitlab CI Runner

GitLab Runner is the tool that is used to run your jobs and send the results back to GitLab.

Gitlab CI Runner

Gitlab CI Runner

Run on:

- Linux
- Docker
- Windows

How to install & configure:

- https://docs.gitlab.com/runner/install/
- https://docs.gitlab.com/runner/register/
- https://github.com/ondrejsika/gitlab-ci-runner

Install Gitlab Runner - Linux

```
sudo wget -O /usr/local/bin/gitlab-runner
https://gitlab-runner-downloads.s3.amazonaws.com/latest/b
inaries/gitlab-runner-linux-amd64

sudo chmod +x /usr/local/bin/gitlab-runner

sudo useradd --comment 'GitLab Runner' --create-home
gitlab-runner --shell /bin/bash

sudo gitlab-runner install --user=gitlab-runner
--working-directory=/home/gitlab-runner

sudo gitlab-runner start
```

Register Gitlab Runner - Linux

```
sudo gitlab-runner register

# or

gitlab-runner register --non-interactive \
   --url $GITLABCI_URL \
   --registration-token $GITLABCI_TOKEN
```

Install Gitlab Runner - Docker

```
docker run -d \
    --name gitlab-runner \
    --restart always \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v /builds:/builds \
    gitlab/gitlab-runner:latest
```

Register Gitlab Runner - Docker

```
docker exec -ti gitlab-runner gitlab-runner register \
   --non-interactive \
   --url $GITLABCI_URL \
   --registration-token $GITLABCI_TOKEN \
   --name $(hostname) \
   --executor docker \
   --docker-image docker:git \
   --docker-volumes '/var/run/docker.sock:/var/run/docker.sock' \
   --docker-volumes '/builds:/builds'
```

https://github.com/ondrejsika/gitlab-ci-runner

Done, check out your Gitlab

First Job

First job

```
git clone git@gitlab.com:test/test.git
cd test
vim .gitlab-ci.yml
git add .
git commit -m "Add CI script"
git push origin master

# .gitlab-ci.yml

job:
  script: echo Hello World!
```

.gitlab-ci.yml

https://docs.gitlab.com/ce/ci/

Jobs

- script
- when
- stages
- only & except
- before_job & after_job
- retry

Script

```
test1_job:
  script: echo 'Run test1 ...'

test2_job:
  script:
    - echo Run 'test2.1 ...'
    - echo Run 'test2.2 ...'
    - echo Run 'test2.3 ...'
```

Stages

```
stages:
  - build
  - test
  - deploy

build_job:
  stage: build
  script: echo 'Building ...'

test1_job:
  stage: test
  script: echo Run test1 ...'

test2_job:
  stage: test
  script: echo Run test2 ...'
```

When

```
cleanup_build_job:
  script: echo Cleanup build when failed ...
  when: on_failure

test_job:
  script: echo Run test ...

deploy_job:
  script: echo Deploy ...
  when: manual

cleanup_job:
  script: echo Full cleanup ...
  when: always
```

Only & Except

```
job:
  # use regexp
  only:
    - /^issue-.*$/
  # use special keyword
  except:
    - branches
```

Variables

- Secret variables are defined in Gitlab
- Some variables set CI runtime
- Public variables are defined in .gitlab-ci.yml

Variables

```
CI
CI_PROJECT_NAME, CI_PROJECT_PATH_SLUG
CI_COMMIT_REF_NAME, CI_COMMIT_REF_SLUG
CI_COMMIT_SHA, CI_COMMIT_TAG
CI_PIPELINE_ID, CI_JOB_ID
CI_REGISTRY, CI_REGISTRY_USER, CI_REGISTRY_PASSWORD
...
```

https://docs.gitlab.com/ce/ci/variables/README.html

Variables

```
variables:
  IMAGE_TAG: myapp:$CI_PIPELINE_ID

job:
  script: docker build -t $IMAGE_TAG .
```

Docker

- Fully supported
- Easiest way how to create build environment
- Easiest way how to run and distribute your software

Docker Environment

```
image: ondrejsika/ci

job:
  image: ondrejsika/ci-go
  script: go build server.go
```

Docker

```
job:
  script:
    - docker build -t $IMAGE .
    - docker push $IMAGE
```

Environments

Environment is used to define that a job deploys to a specific environment.

If environment is specified and no environment under that name exists, a new one will be created automatically.

Environment

```
deploy:
  script: echo 'Deploy!'
  environment:
    name: $CI_COMMIT_REF_SLUG
    url: https://$CI_COMMIT_REF_SLUG.example.com
```

Deployments

- Automatic
- Manual

Auto vs Manual Deployments

```
auto_deploy_job:
  script: echo Auto Deploy!
  environment:
    name: deployment-$CI_PIPELINE_ID

manual_deploy_job:
  when: manual
  script: echo Manual Deploy!
  environment:
    name: deployment-$CI_PIPELINE_ID
```

Stop Deployment

```
deploy_job:
  stage: deploy
  script: echo Deploy!
  environment:
    name: deployment-$CI_PIPELINE_ID
    on_stop: stop_deploy_job

stop_deploy_job:
  stage: deploy
  script: echo Stop!
  when: manual
  environment:
    name: deployment-$CI_PIPELINE_ID
    action: stop
```

Try it!

https://github.com/ondrejsika/linuxdays2018

https://gitlab-demo.xsika.cz/demo/linuxdays2018

Resources

- https://about.gitlab.com/features/gitlab-ci-cd/
- https://docs.gitlab.com/ce/ci/
- https://docs.gitlab.com/ce/ci/yaml/
- https://docs.gitlab.com/ce/ci/quick_start/

- https://ondrej-sika.cz/blog/2018/gitlab-ci-docker-linuxdays/
- https://github.com/ondrejsika/ondrejsika-ci-docker
- https://github.com/ondrejsika/traefik-ssl
- https://github.com/ondrejsika/gitlab-ci-runner

Thank you & Questions

Ondrej Sika

email: ondrej@ondrejsika.com
web: https://ondrej-sika.cz
twitter: @ondrejsika
linkedin: /in/ondrejsika/

https://sika.link/linuxdays2018