DAY OF DOCKER OSL 2015

DECEMBER 1st

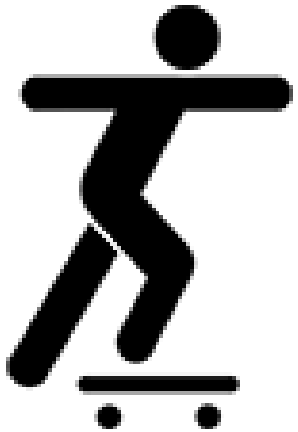
It's whales all the way down...

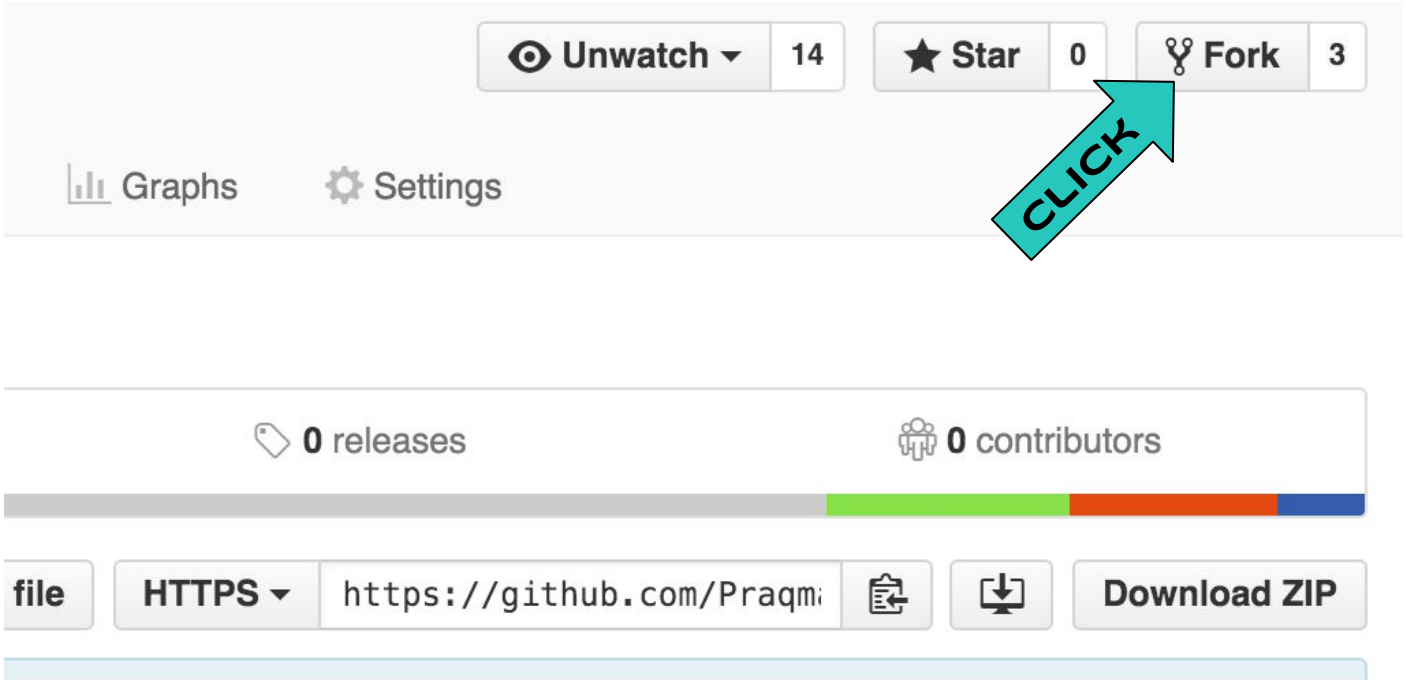
Going "all in" with Docker in your continuous delivery setup

Getting your host ready

- Fork our github repository to your local github account. (https://github.com/Praqma-training/dayofdocker15)
- Login to your Zetta cloud instance (Docker Host) using SSH and the IP and credentials. (*Credentials will be handed over to you shortly.*)
- Clone your repo to your docker host:
  - $ git clone https://github.com/<YOUR USER>/dayofdocker15.git
- And then:
  - $ cd dayofdocker15/containers
  - $ docker-compose up
  - $ docker-compose start

Fork Github repo



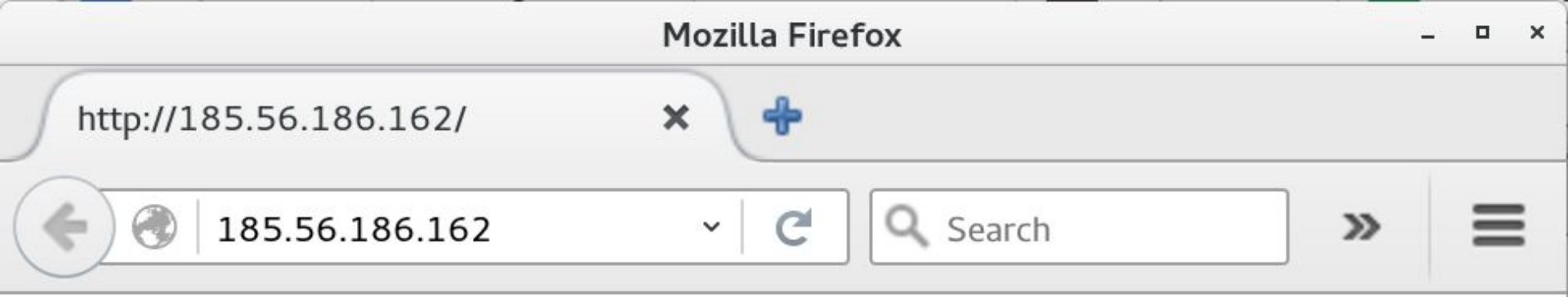


Getting your host ready

- `$ ssh student@<your machine ip>`
  - password: **`P@r@c3t500m9`**

Then on cloud host:

- `$ git clone https://github.com/<YOUR_USER>/dayofdocker15.git`
- `$ cd dayofdocker15/containers`
- `$ docker-compose up`
- `$ docker-compose start`



Welcome to Day of Docker 2015

Jenkins is at : http://YOURHOST/jenkins
Artifactory is at: http://YOURHOST/artifactory
Docker Registry is at: http://YOURHOST/registry (You will see a blank page! Not much helpful. ;)
The Go Web Server (you will be compiling) will be at: http://YOURHOST:8000

* Replace YOURHOST with the IP of your DockerHost.

Exercise - generate the pipeline

- Step 1 - Create a seed job
- Step 2 - run the provided JobDSL
- Step 3 - *there is no step 3*
- Step 4 - *profit*

Step 1 - Create seed job

… add parameter for GITHUB_USERNAME

Source Code Management

- None
- CVS
- CVS Projectset
- Git

Repositories

Repository URL: https://github.com/drBosse/dayofdocker15.git

Then:

jobDSL/*.groovy

Save and run


Jenkins
Jenkins
seed-jan
Back to Dashboard
Status
Changes
Workspace
Build Now

Congrats

You should now have:

- 3 jobs:
  - server-build
  - server-test
  - server-release
- A ‘View’ (tab)
- A Build pipeline view

A note on versioning

The simplified story:

- Semantic versioning
- Controlled by developer
- In `version.txt` file
  - Pulled from repo in build phase
  - Passed through all the way to release phase
  - Used to tag release version of docker image

Build phase - simplified

```
docker build -t drbosse/http-app:snapshot .
```

- *build Gonah and tag with snapshot*

```
docker run -d --name testing-app -p 8001:8000 drbosse/http-app:snapshot
```

- *run Gonah on test port*

```
docker run --rm siege-engine -g http://<ip-of-http-app-container>:8000/
```

- *Use Dockerized **siege** to test that the server responds*

If all is OK, we trigger the test phase, and pass the image id.

NOTE: Look at the full version in your own “build-browser” job

Testing phase

“Deploy to test”

```
docker run -d --name testing-app -p 8000:8000  $IMAGEID
```

- *Run test version on port 8000*

“Run functional test”

```
docker run --rm siege-engine http://<ip-of-http-app-container>:8000/
```

- *Load test with siege engine*
- *If availability is OK, then tag image as stable*

```
docker tag $IMAGEID drbosse/http-app:stable
```

- *and for fun, we plot some of the output from siege*
- *If everything is OK, we call trigger a release*

Release phase

Tag with version nr. and ‘latest’

```
docker tag -f drbosse/http-app:stable drbosse/http-app:latest
docker tag -f drbosse/http-app:stable drbosse/http-app:$VERSION
```

Deploy to “production”

```
docker run -d --name deploy-app -p 81:8000 drbosse/http-app:latest
```

- *Run prod on port 81 to avoid conflict with existing Apache*

- If everything is OK, we are just happy