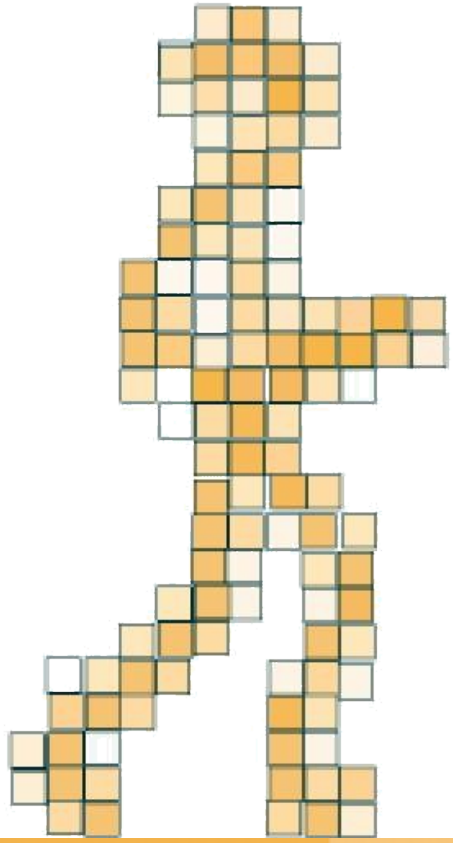
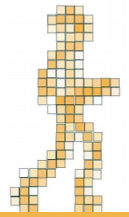


Continuous Deployment with Jenkins for Spring Boot and Angular with Docker

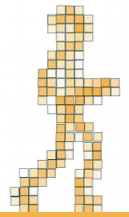
Who am I?

- Jonas Andersen
  - B.Sc. (Hons) Computer Science
- 2007 – present - Senior Consultant at aragost Trifork ag based in Zurich
  - Software development and agile coaching services
- 2000 – 2007 - IT Specialist & IT Architect at IBM Global Services based in Copenhagen



Set up local Git server (gogs)

- Image on Docker hub https://hub.docker.com/r/gogs/gogs/
- GitHub page https://github.com/gogits/gogs/tree/master/docker



Set up local Git server (gogs)

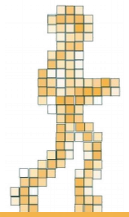
```
$ docker pull gogs/gogs
$ docker volume create --name todo-gogs-data
$ docker run -d --name todo-gogs -p 3000:3000 -p 3022:22  -v todo-gogs-data:/data gogs/gogs
```

Open browser to http://localhost:3000

Database Settings
- Database type: SQLite3
- Path: /data/gogs.db (add / prefix)

Application General Settings
- SSH Port: 3022

Admin Account Settings
- Username: gituser
- Password: gituser



Set up local Git server (gogs)

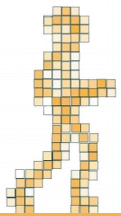
Open browser to http://localhost:3000

Login as gituser / gituser

Select the + sign and select „New Migration“

Migrate https://github.com/aragost-ja/zh-microservices-springdocker-svc.git as todo-svc

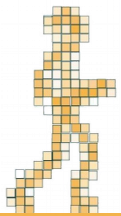
Migrate https://github.com/aragost-ja/zh-microservices-springdocker-web.git as todo-web



Set up Jenkins

Jenkins homepage
https://jenkins.io/index.html

Official Jenkins Docker image
https://github.com/jenkinsci/docker/blob/master/README.md



A custom Jenkins image

$ git clone https://github.com/aragost-ja/zh-microservices-springdocker-jenkins.git
$ cd zh-microservices-springdocker-jenkins
$ docker pull jenkins/jenkins:lts
$ docker build -t todo-jenkins:1 .
$ docker volume create --name todo-jenkins-data
$ docker run -d --name todo-jenkins -p 3080:8080 -p 50000:50000 \
-v /var/run/docker.sock:/var/run/docker.sock -v /usr/bin/docker:/usr/bin/docker \
-v todo-jenkins-data:/var/jenkins_home todo-jenkins:1

Open browser to http://localhost:3080/jenkins

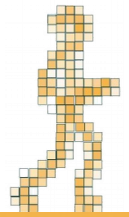
$ docker logs todo-jenkins

Copy the admin password from the logs

Select the „Install suggested plugins“ option

Create a new user admin / admin123

Select restart (the browser window may not refresh correctly, do an explicit reload)



A custom Jenkins image

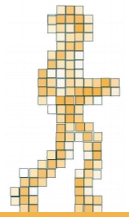
Select Manage Jenkins > Global Tool Configuration

Expand the Maven section and add a new Maven installer
- Name: Maven 3.3.9 (important, this is the name used in the Jenkinsfile to reference the tool)
- Install automatically checked
- Install from Apache, Version 3.3.9 selected

Expand the NodeJS section and add a new NodeJS installer
- Name: Node 6.x (again, the name is referenced in the Jenkinsfile)
- Install automatically checked
- Install from nodejs.org, Version NodeJS 6.11.2 selected
- In „Global npm packages to install“ enter „@angular/cli“

Leave the Docker section unconfigured. We have already added Docker to this container through the custom image and mounts.

Select Save

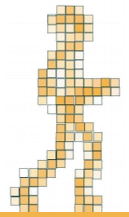


The application

A Spring Boot 1.5 based backend providing a REST API

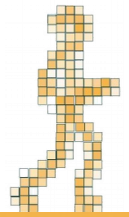
An Angular 4 based frontend providing a Single Page Application as User Interface to the REST API

(A note on environments – dev vs „production“ and the nginx reverse proxy part) – dev is using „ng serve“ production is build used by Jenkins



The Spring Boot Backend

- Created with Spring Tool Suite 3.9
- Building the application with Maven
- Using the `openjdk:8-jdk-alpine` as the base Docker image https://hub.docker.com/_/openjdk/
- Spring Guide on Spring Boot and Docker https://spring.io/guides/gs/spring-boot-docker/



Backend pipeline project

Open browser to http://localhost:3080/jenkins and log in as admin / admin123

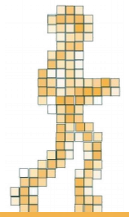
Select „create new jobs“ link

Enter „todo-svc“ in the name field

select Pipeline and select OK

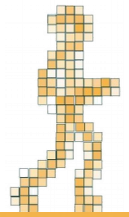
Go to the Pipeline section and change the Definition to „Pipeline script from SCM“
Select Git as SCM
Set Repository URL to http://172.17.0.1:3000/gituser/todo-svc.git (replace IP as necessary)
Add credentials for the gituser, ID must be gogs-gituser (important!)
Uncheck „Lightweight checkout“

Select the „Build Now“ link on the project page



The Angular frontend

- Created with angular-cli (ng new)
- Packaging the „dist“ directory in a nginx based Docker image
- Nginx image: https://hub.docker.com/_/nginx/



Frontend pipeline project

Open browser to http://localhost:3080/jenkins and log in as admin / admin123

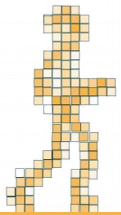
Select „create new jobs“ link

Enter „todo-web“ in the name field

select Pipeline and select OK

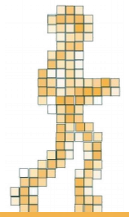
Go to the Pipeline section and change the Definition to „Pipeline script from SCM“
Select Git as SCM
Set Repository URL to http://172.17.0.1:3000/gituser/todo-web.git (replace IP as necessary)
Select credentials for the gogs-gituser
Uncheck „Lightweight checkout“

Select the „Build Now“ link on the project page



CR1 – docker containers

- In Gogs, merge the CR1 into the master branches
- Build the projects in Jenkins



CR2 – tag the Git repository

Select Manage Jenkins > Configure System

Under Global properties, check „Environment variables“ and add a variable named „DOCKER_GATEWAY_IP“ with the value of your Docker host IPv4 address (/sbin/ifconfig docker0)

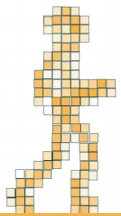
This variable is referenced in the Jenkinsfile script for the push of Git tags to the origin (GOGS server)

Set the global Git user properties in the container:

$docker exec -u jenkins todo-jenkins git config --global user.name "gituser"
$docker exec -u jenkins todo-jenkins git config --global user.email "gituser@localhost.localdomain"

Merge the CR2 branches into master in Gogs

Build the projects in Jenkins

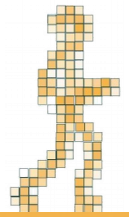


CR3 – New feature: Add new todo

We‘re adding a new feature to the application. The user can now add new ToDo items to the list.

Merge the CR3 branches into master in Gogs

Build the projects in Jenkins



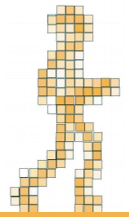
CR4 – Deploy staging and production containers

We will now run the created images.

We define containers for the frontend and backend container for a staging and a production environment.

Merge the CR4 branches into master in Gogs

Build the projects in Jenkins

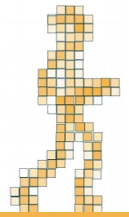


CR5 – Approval of production deployments

Automatic deployment to production is not desired. We want to explicitly allow this.

Merge the CR5 branches into master on Gogs

Build the projects in Jenkins

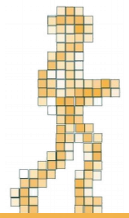


CR6 – New Feature: Edit existing todo

New feature – users can now update existing ToDo items.

Merge the CR6 branches into master in Gogs

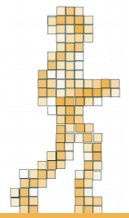
Build the projects in Jenkins



Error in Production!

The edit functionality is no longer working! The User Interface displays the edit popup but saving the changes does not work.

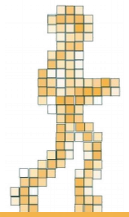
What happened?



CR7 – Milestones and locks

Merge the CR7 branches into master in Gogs

Build the projects in Jenkins



CR8 - housekeeping

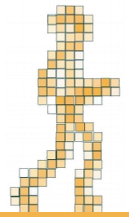
We are generating new images with each build. Let‘s clean up the old images.

(using LABEL in Dockerfile)

Remove the old images manually

Merge the CR8 branches into master in Gogs

Run 2-3 builds and verify that only the currently used image is listed



Caveat emptor

- The Pipeline Scripts are basic examples – be sure to understand them before using in the your environment! (e.g. the „input“ element is still executed with a node allocated)
- The example applications provide basic examples. Shortcuts have been taken which are not appropriate for production use!
- Beware of resource issues when using Java inside Containers - https://developers.redhat.com/blog/2017/03/14/java-inside-docker/