



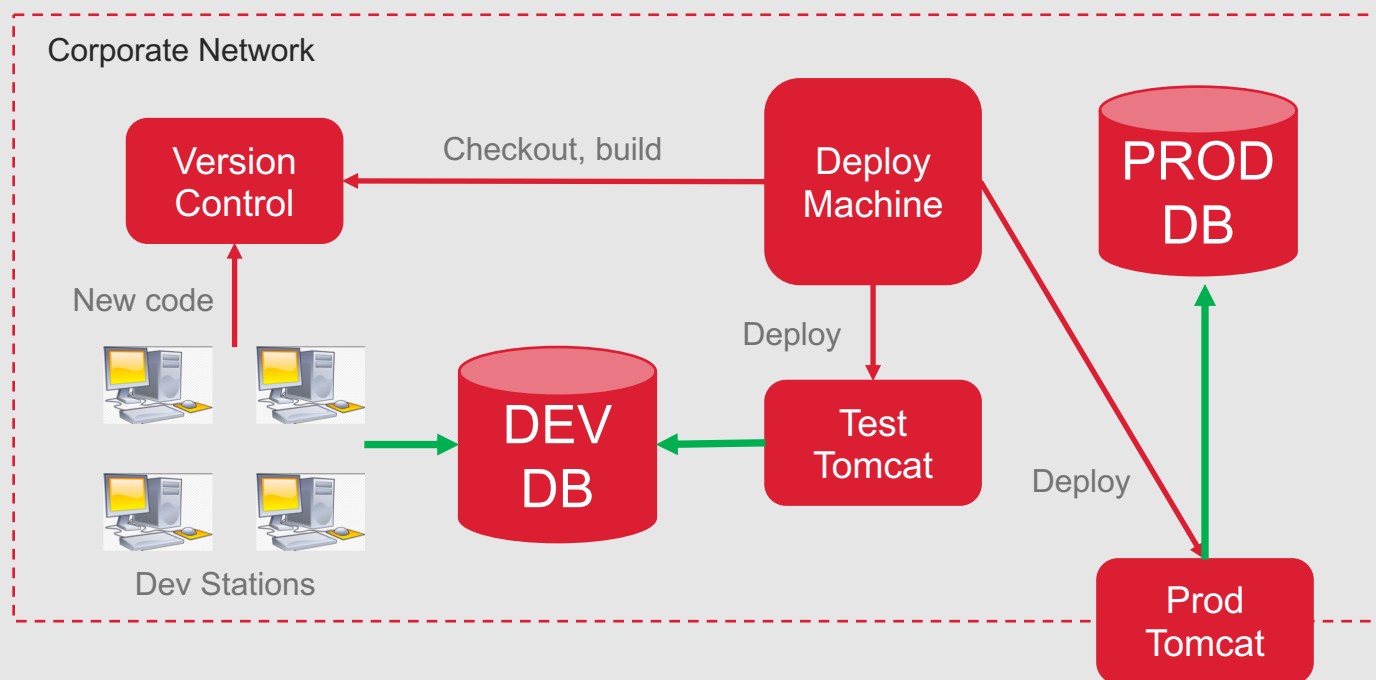
# DOCKER CONTAINERS

Developer's experience and building robust developer environments





# The Old Way





# We Were Lucky

It was not possible to install used  
database engine to  
a developer machine

(at least it was not easy)





# Years Went On...

We investigated on building  
*batteries included* developer environments  
with stuff like VirtualBox and Vagrant





# Enter the Docker







## spin-dev-dbs.sh

```
#!/bin/bash

cd "$(dirname "$0")"

echo "Spinning up postgres in docker. Hold your horses!"

if [ ! "$(docker ps -a -q -f 'name=acme-api-postgres')" ]; then
    echo "Container does not exists, creating it"
    docker run --name acme-api-postgres -e POSTGRES_PASSWORD=password -p 5432:5432 -d postgres:10
    echo "Sleeping for 20 s ensure postgres responds... After that, creating databases..."
    sleep 20
    docker run -e "PGPASSWORD=password" -i --rm --link acme-api-postgres:postgres postgres \
        psql -h postgres -U postgres -c "CREATE DATABASE acme_api"
    docker run -e "PGPASSWORD=password" -i --rm --link acme-api-postgres:postgres postgres \
        psql -h postgres -U postgres -c "CREATE DATABASE acme_api_test"
else
    echo "Starting already initialised container"
    docker start acme-api-postgres
fi
```

## psql-in-docker.sh

```
#!/bin/bash

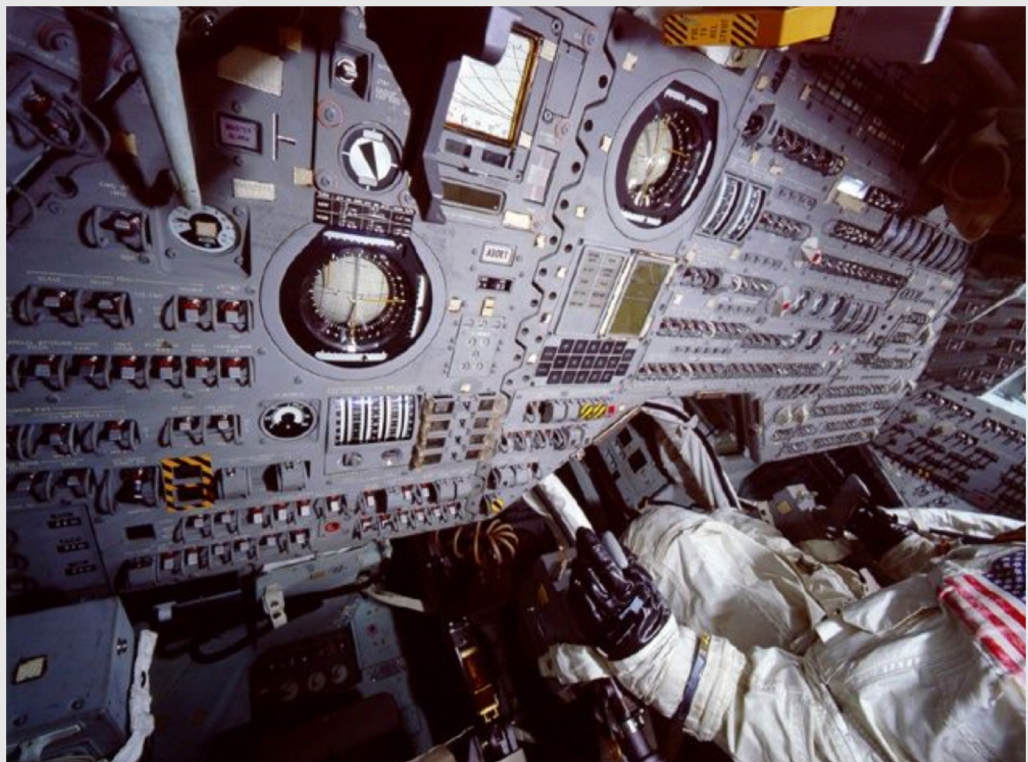
DB_NAME=$1

docker run -e PGPASSWORD=password -it --rm --link acme-api-postgres:postgres postgres \
    psql -h postgres -U postgres $DB_NAME
```















Docker is like a spacecraft

Containers should be like screwdrivers





A Container is an abstraction of  
**Unit of Work**



# Mixing Building and Running in the Same Container

```
FROM node:8

WORKDIR /usr/src/app
COPY package*.json ./

RUN npm install

# Bundle app source
COPY . .

EXPOSE 8080
CMD [ "npm", "start" ]
```

This is a really common pattern, also in production systems

It produces a Docker image that is more like a spacecraft than a screwdriver

CMD is just a default for Docker run command

So one can easily run something like:  
`docker run -it acme-api npm login`



# Multi Stage Builds is a Bliss

```
FROM node:10.4-alpine

RUN mkdir /app
WORKDIR /app
RUN npm -v

COPY . .

RUN npm install
RUN npm run tsc
RUN rm -rf node_modules
RUN npm install --only=prod

FROM node:10.4-alpine

RUN mkdir /app
WORKDIR /app

COPY --from=0 /app/node_modules node_modules
COPY --from=0 /app/out .

EXPOSE 3000

CMD [ "node", "index.js"]
```

I think this is the most important new feature in Docker, because this is something that was really complex to achieve before.

Unfortunately all official node alpine images include NPM. There is a Github issue on this:

<https://github.com/nodejs/docker-node/issues/404>

But now your image starts to look more like a screwdriver!



# But what you can and should do..

... is to create "incubator" containers that exist just for built just for building you app image

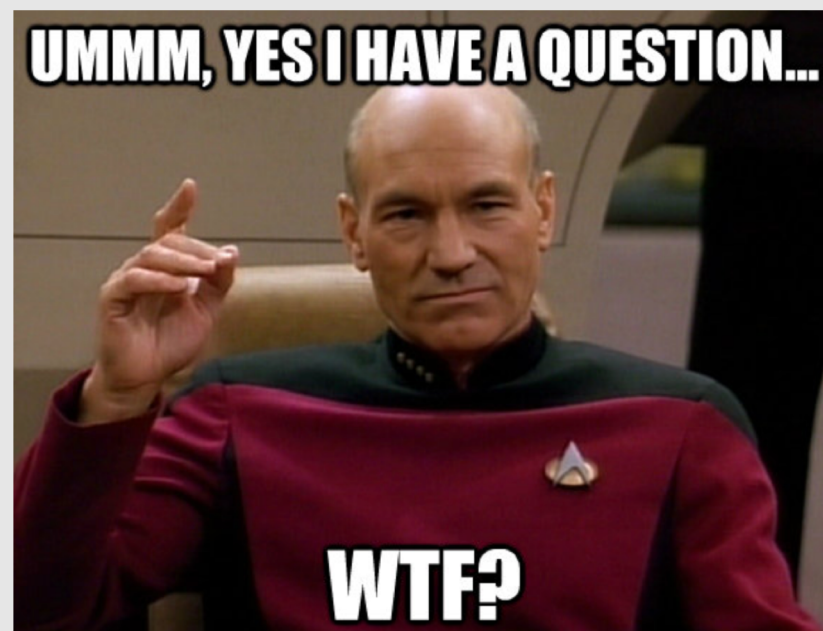
The best situation would be that one is able to run incubator locally

# Avoid Building Hard Environmented Images

```
docker run acme-api-production:1.2.3
```

```
docker run acme-api-development:1.2.3
```

```
docker run acme-api-local:1.2.3
```





# Build Flexible Images

```
docker run \  
  -e "DB_HOST=localhost" \  
  -e "DB_USER=foobar" \  
  -e "DB_PASS=notsecret" \  
  acme-api:1.2.3
```

```
docker run \  
  -e "DB_HOST=db.acme.org" \  
  -e "DB_USER=acme_db" \  
  -e "DB_PASS=crypto-store:/acme-api/db-password" \  
  acme-api:1.2.3
```



Make tooling! (run-acme-api-locally.sh)

Sensible local defaults!

Devs need to be able to easily launch same containers locally!

# I want To Hot-Reload Code From My IDE to a Running Container Locally!?



Don't **EVER** support this in the image that is meant to be deployed to a live environment

Instead, make tooling for building local images that support hot reloading

```
docker run acme-api-hotreload:localbuild
```

(Disclaimer: I don't know, if this is actually a good idea in the first place...)



Give team the possibility to embrace  
Docker and Containers





We should talk more about  
*Development Architecture*

Running in production is a one thing

Developing is something that has not  
been addressed that much



*Works on my machine*

We must avoid magic machines

Your repository should always be runnable  
in a new machine "in fifteen minutes"



If you have a good workflow

Share it, if possible

Current blogsphere on  
(cloud)

development architectures  
is somewhat non-existent



# Embrace DevOps Culture

Do not hide containers

Embrace and enable developers to  
test and run stuff locally in containers  
as well



# Tänaan!!!

**JUHO FRIMAN**

Software Architect

[juho.friman@solita.fi](mailto:juho.friman@solita.fi)

[@commafakir](#)

Ps. We are hiring!



