Build and Development Environments with Nix and Docker

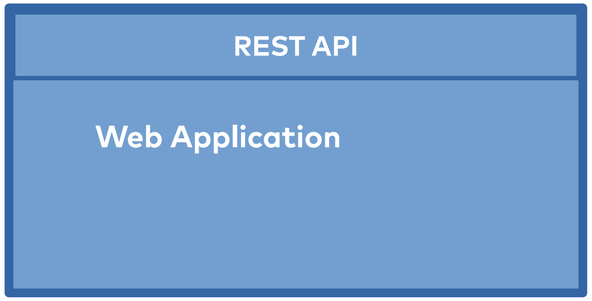
Christine Koppelt
christine.koppelt@innoq.com

Linuxwochen Wien 2018

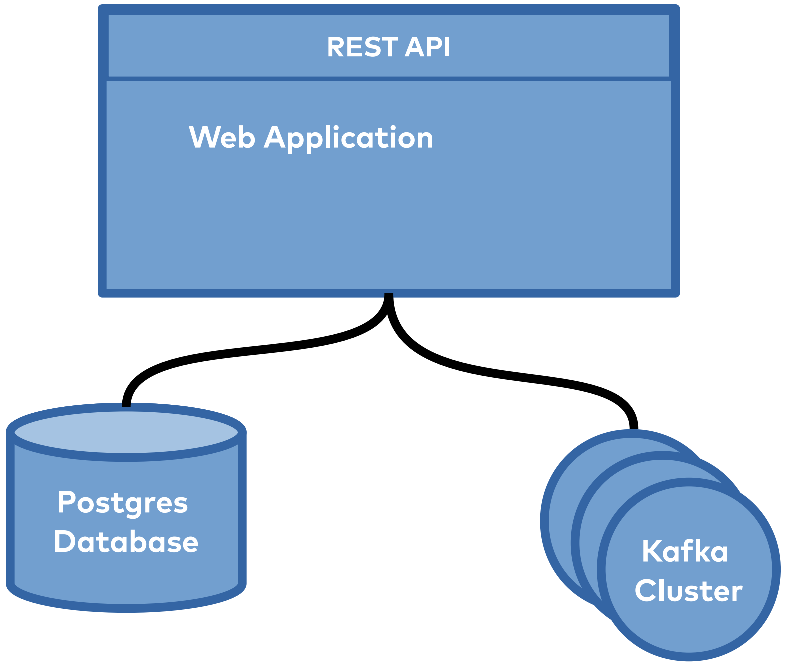
About Me

- Software Developer for 10 years
- Senior Consultant at INNOQ
- Regularly working with Docker
- Using NixOS in my free time for ~2 years
- Started using Nix in commercial projects some months ago

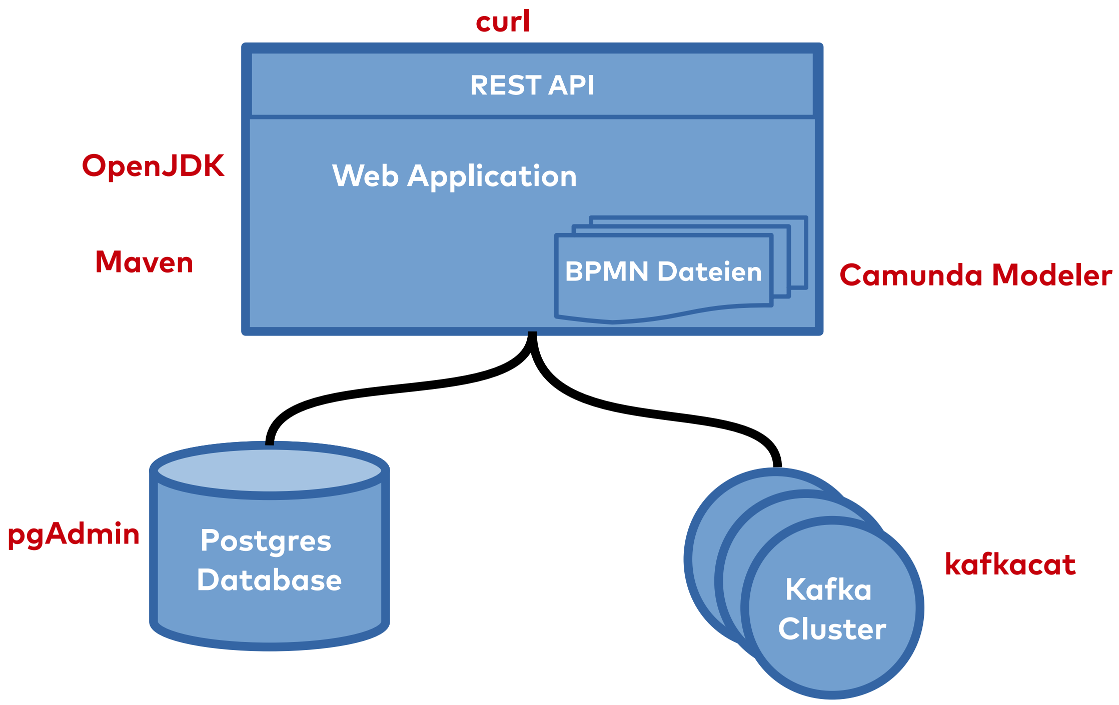
Developing an application …



… often requires some infrastructure services …



… and a lot of development tools



Working on more than one application …

- Use the same service versions for develoment and production
- Tools and services need to be available in multiple versions
- Hassle-free switching between projects

Working in a team …

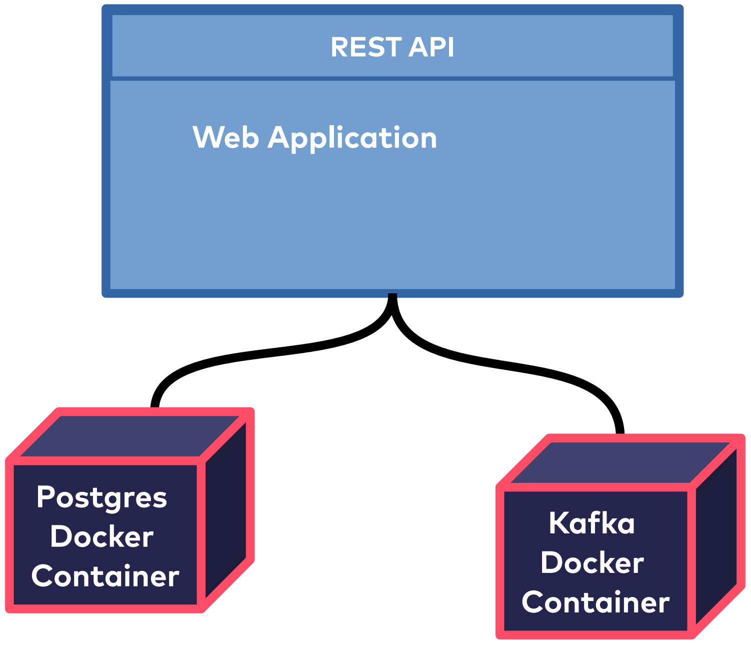
- Default: Every team member works with the same versions
- Environment should be reproducible & updatable
- Fast start for new developers

Not so good Options

- Manual installation
- Package manager of your Linux distribution
- Programming language specific package managers
- Hand-written scripts

Step 1:
Automate Services Setup Using Docker

Goal


REST API
Web Application
Postgres Docker Container
Kafka Docker Container

Docker in a Nutshell

- Software can be installed & started inside separated „boxes“ called containers
- Central repository with premade containers
- Open-Source, available for Linux, Mac and Windows
- Provides uniform interface for starting applications

Example: Running PostgreSQL

```
docker run -d \
           -e POSTGRES_PASSWORD=secret \
           -p 5432:5432  \
           postgres:10.3
```

Script it with Docker Compose

stack.yml

```
version: '3.1'
services:
  db:
    image: postgres:10.3
    restart: always
    environment:
      POSTGRES_PASSWORD: secret
  kafka:
    image: ...
```

INNOQ

```
docker-compose -f stack.yml up
```

Benefits

- ✓ Scripted, versionable & reproducible
- ✓ Setting up multiple services in one step
- ✓ Isolated, doesn't affect operating system
- ✓ Multiple service versions in parallel
- ✓ Keep versions in sync within the development team
- ✓ … and with the Continuous Integration & production servers

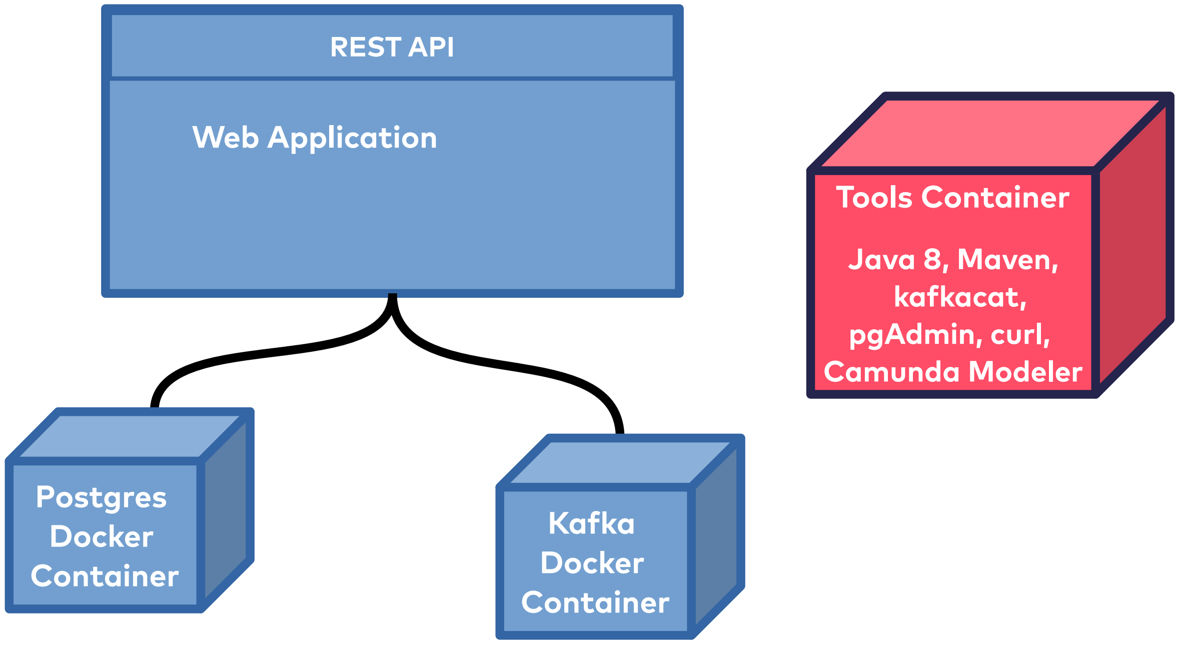
Caveats

- ✗ Custom-made images are somewhat cumbersome to manage

Step 2:
Automate Tooling Setup Using Docker

INNOQ

A good idea?


REST API
Web Application
Postgres Docker Container
Kafka Docker Container
Tools Container
Java 8, Maven, kafkacat, pgAdmin, curl, Camunda Modeler

Approach

- Tools are installed within the container
- Mount your local src directory into the container
- Call the tool within the container

Example: Running Maven (basic version)

```
docker run -it --rm    \
    -v "$(pwd)":/usr/src/mymaven \
    -w /usr/src/mymaven \
    maven:3.3-jdk-8 \
    mvn clean install
```

It becomes only more ugly

- Graphical tools
- User permissions
- Caching files

No cool solutions

- Aliases?
- Develop completely within the container?
  - SSH Shell or Shell via Docker exec
  - Graphical tools?

Benefits

- ✓ Setting up multiple tools in one step
- ✓ Isolated, doesn't affect operating system
- ✓ Multiple tool versions in parallel
- ✓ Keep versions in sync within the development team

Caveats

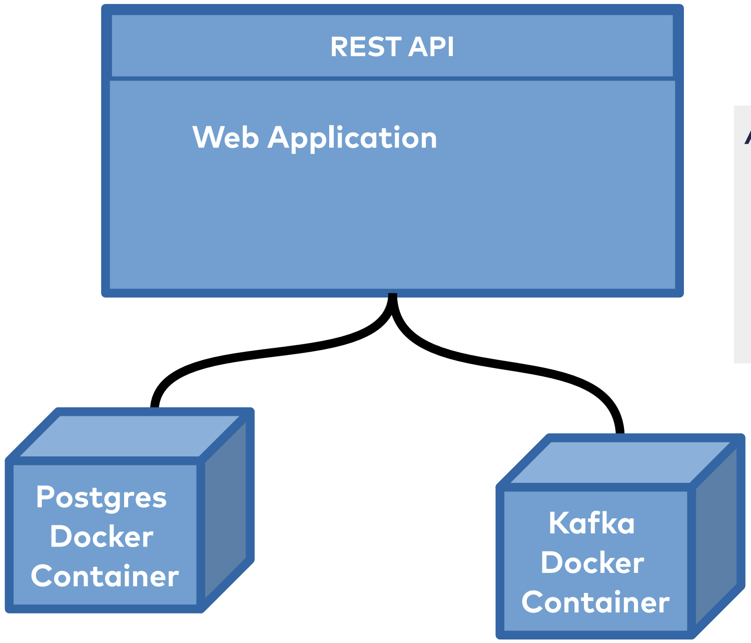
- ✗ Ugly command line calls
- ✗ Adding new tools to the Docker image needs a rebuild of the Docker image
- ✗ Graphical tools even more cumbersome

Step 2: Alternative Automate tooling Setup Using Nix

What is Nix?

- Package manager
- Contains a broad range of tools
  - ~13.000 packages
  - Own packages can be added
- Own configuration language
- Works on Mac and Linux
- Immutable package store, multi-version support

Stored separately



```
/nix/store
    4k3ah-openjdk-9.0.4
    15Jns-maven
    3Byd1-kafkacat
    ghlhk-pgAdmin
    Lsn08-curl
    jicnp-camunda_modeler
```

INNOQ

Loading tools on the fly

nix-shell -p a_package

```
ck@ck-innoq:~/myproject$ java -version
openjdk version "1.8.0_131"
ck@ck-innoq:~/myproject$ nix-shell -p openjdk9 maven
[nix-shell:~/myproject]$ java -version
openjdk version "9.0.4-internal"
```

What happens

- Downloads packages
- Stores them at /nix/store

  Example:

```
/nix/store/2fiavk609lgb9wsr560lkjf6wyx7d9a3-apache-maven-3.5.2
```

- Sets Links

```
[nix-shell:~/Dokumente/microxchg]$ which mvn
/nix/store/2fiavk609lgb9wsr560lkjf6wyx7d9a3-apache-
maven-3.5.2/bin/mvn
```

Write a default.nix script

```
with import <nixpkgs>{};
stdenv.mkDerivation {
   name = "my-service";
   buildInputs = [openjdk9 maven
                  kafkacat curl];
}
```

Loading configuration

```
nix-shell default.nix
```

Define new package (schematic)

```
camunda_modeler = stdenv.mkDerivation {
  name = "camunda_modeler";
  src = pkgs.fetchurl
          { url = "https://..."; sha256 = "..."; }
  installPhase =
    ''
       tar -xzf $src
    '';
};
```

Add it to buildInputs

```
stdenv.mkDerivation {
 name = "my-service";
 buildInputs = [openjdk9 maven
                kafkacat curl
                camunda_modeler];
}
```

Version Pinning

```
let
  hostPkgs = import <nixpkgs> {};
  nixpkgs = (hostPkgs.fetchFromGitHub {
    owner = "NixOS";
    repo = "nixpkgs-channels";
    rev = "9c31c72cafe536e0c21238b2d47a23bfe7d1b033";
    sha256 = "0pn142js99ncn7f53bw7hcp99ldjzb2m7xhjrax00xp72zswzv2n";
  });
in
with import nixpkgs {};
```

Configure Tools

```
with import <nixpkgs>{};

let curl = pkgs.curl.override {
  zlibSupport   = true;
  sslSupport    = true;
  http2Support  = false;
};
in
stdenv.mkDerivation {
  name = "my-service";
  buildInputs =  [ openjdk9 maven kafkacat curl camunda_modeler ];
}
```

Benefits

- ✓ Low overhead
- ✓ Setting up multiple tools in one step
- ✓ Hardly affects host system
- ✓ Multiple tool versions in parallel
- ✓ Keep versions in sync within the development team

Combination of Docker & Nix

- Docker
  - Fast development setup for services like message broker, databases and custom services
- Nix
  - Setup of development tools like custom editors, database & messaging clients, networking tools

More information about Docker

- Official documentation

  https://docs.docker.com/

- Central container image hub

  https://hub.docker.com/

More information about Nix

- Official Website

  https://nixos.org
- My Twitter Account

  @nixos_muc
- Meetups

  Europe: Munich, Berlin, Amsterdam, London

Questions?

Christine.Koppelt@innoq.com