Docker 在开发中应用二三例

—— 程序猿必备开发技能

About Me

- 中文名: 朱礼
- 常用ID: FooTearth footearth@gmail.com
- 受开源熏陶多年，技术上比较爱折腾
- 重度依赖 google && github && docker
- 目前在武汉一家创业公司做架构，现阶段以 node 生态下的前端为主
- 封了一些常用语言环境的镜像，并在开发中使用

基础设施：两个典型场景快速体验

docker

场景一：搭建 VPN

- 场景简介
  - GooGle
    - 程序猿必备开发技能
    - 设计师都会使用的技能
- Source:
  https://github.com/Mooxe000/mooxe-docker-ladder

场景二：搭建 GitLab && Jenkins

- 场景简介
- Source：
  - GitLab：https://github.com/sameersbn/docker-gitlab
  - Jenkins：https://github.com/jenkinsci/dockero

为什么从开发这个角度讲

- 团队内部推广比较困难
- 多数讲题背景生态讲的多，实操讲的比较少

企业内推广不畅

原因一：对新技术的尝试多半会比较谨慎

- 开发人员：听说过，没试过，不知道能干嘛
- Leader：觉得既没时间又没人探索
- CTO：认为风险大

？什么样的新技术容易被接受？

什么样的新技术容易被接受

- 曝光率高
- 无副作用

曝光率高

火到不知道就不好意思跟人聊天

- React
- Docker

无副作用

- 基础设施
- 开发环境
- 测试环境

企业内推广不畅

原因一：没有明确的职位，职责不明

- 开发认为属运维，运维认为属开发
- DevOps

DevOps 现状

- 拉钩搜索
  - 北京 2 页
  - 上海 1 页
  - 深圳 1 个
  - 广州 1 个
  - 杭州 2 个
  - 武汉 0 个

DevOps 阶段

- 工具团队（提供基础设施、封开发镜像）-- 开发
- 应用 Docker 化（容器化） -- 开发、测试
- 持续集成持续交付（CICD） -- 测试、生产

开发环境：以 web 前后端开发 为例，动手 构建开发镜像

- base 镜像
- 基于 base 的各语言 开发环境 镜像

base 镜像

- Source:
  https://github.com/Mooxe000/mooxe-docker-base
- 自定义的环境: 各团队，各运维人员 风格 习惯 不通
  - 指定 操作系统及版本（centos / ubuntu / arch / (gentoo/lfs) / Apline）
  - 系统、软件包更新到最新
  - 指定 shell 环境（bash / zsh / fish / elvish）
  - 更新源，host，dns 等

基于 base 的各语言 开发环境 镜像

- nodejs
  - Source:
    https://github.com/Mooxe000/mooxe-docker-node
  - npm - nvm - cnpm
- php（未更新到最新）
  - Source:
    https://github.com/Mooxe000/mooxe-docker-php
  - php - composer - 国内加速
  - PS. 案例
    - 一次关于 要不要使用 Docker 的争论
    - 生产环境故障，查了一天查下来，发现 php 版本被升级
- golang
  - Source
    https://github.com/Mooxe000/mooxe-docker-golang
- python
  - Source
    https://github.com/Mooxe000/mooxe-docker-python
  - pip-pyenv-virtualenv

测试环境：基于 docker 的 zookeeper - activemq 测试环境

- 项目介绍
- source:
  - zookeeper
    https://github.com/Mooxe000/mooxe-docker-zookeeper
  - activemq
    https://github.com/Mooxe000/mooxe-docker-activemq
- 使用 golang 作为 系统脚本语言
  - 为什么选择 golang

Tips

如何进入一个已经运行的docker

- 过去很麻烦，借助脚本 或 第三方工具
- docker exec（不确定是哪个版本加上的）

Tips

docker 镜像的保存和加载（save && load）

Tips

开发中，推荐 mount 本地源代码目录。

- 分离 代码 和 运行环境
- 原因，假设 mount 源代码到 容器中 固化
    - 改代码 推一次，再拉一次
    - 如果改动了构建，还得得 重新构建 镜像

Tips

docker-osx-dev

- 挂载体积过大，响应超时（ls）EX. 编译 android 内核
- 前端 hot-reload 失效。EX. webpack hot-reload
    - 临时解决办法：watch 之后 触发 读写该源文件
        - 副作用：光标每次定位到文件开始位置
- Source: https://github.com/brikis98/docker-osx-dev

Tips

docker-machine-nfs

osx 上 跑 gitlab 时，拉 postgresql 镜像 会报没有权限 的 错误

```
brew install docker-machine-nfs
# if docker_machine_name is 'dev'
docker-machine restart dev
docker-machine-nfs dev --nfs-config="-alldirs -maproot=0"
```

Tips

探讨 docker file

- 可编程性不强 导致 可复用能力不强
- show demo
  - 使用 node flightplan 参照 base 镜像 初始化 服务器
- 可尝试探索的方向
  - 预编译 生成 dockerfile
  - 绕过 dockerfile，直接调用 api 执行（可行性）
  - tasks 通过容器 编译 二进制执行文件（like golang），再送到容器 中执行

在开发环节 docker 的优势

- 环境搭建 && 开发 分离
  - 一次固化，随时随地使用，而不用关心环境如何搭建
- 环境一致
  - 所有开发人员的运行环境都是一致的
  - 引申 到 测试、生产环境
- 系统隔离
  - 应用运行环境不再与宿主环境相关
- 运行环境隔离
  - 多运行环境相互不受影响
- 服务和数据隔离
  - 协作不共用 服务，避免 数据污染

所有示例都工程固化发布于 GitHub 的开放项目

- PPT - https://github.com/Mooxe000/MeetUp/

固化

- 何为固化
    - 文本化
        - 可版本控制
        - 可追溯
    - 易恢复
- 企业环境中 应 固化 Everything
- 固化服务器环境只是其中一个环节

联系方式

- GitHub
  - https://github.com/footearth
  - https://github.com/Mooxe000
- WeChat
  - footearth
- Email
  - footearth@gmail.com

谢感大家

感谢以下服务或者项目

GooGle

GitHub

Docker

DaoCloud

NodePPT