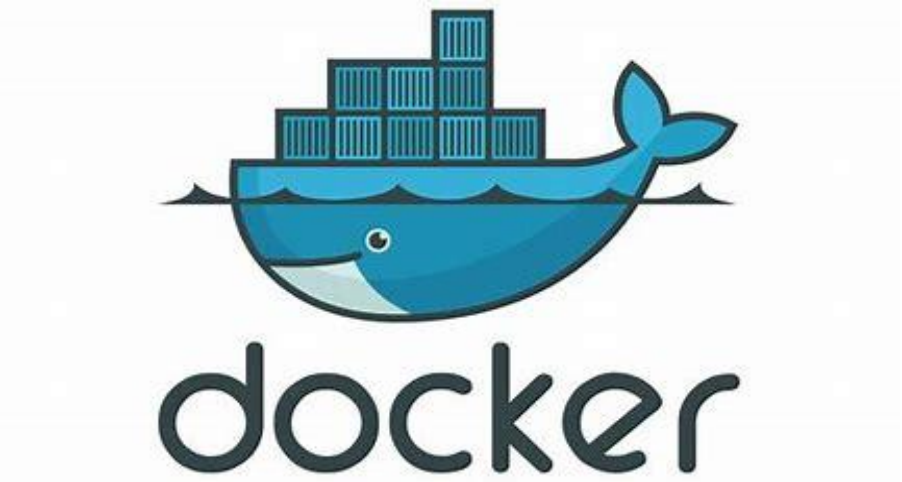
学了Docker，妈妈再也不用担心我的网站部署了！

寒假训练项目-Week1

清华大学自动化系学生科协



主讲人：张竞择

日期：2023.1.17

目录

目录

Docker基本概念

- **Docker**
  - ~~一只驮着很多集装箱的鲸鱼~~
  - Docker是一个用于开发、配置、运行的开放平台。Docker将应用与环境相隔离，让应用程序可以得到快速交付。
  - 通俗地讲，Docker是一个虚拟环境容器，可以将你的开发环境、代码、配置文件一并打包放到这个容器中，并发布和应用到任意平台中。

Docker基本概念

- **What can Docker do?**
  - 开发：Docker可以当成轻量级的虚拟机，拥有独立的环境；折磨人的科研环境配置的过程使用docker可以大大简化
  - 部署：利用Docker可以将程序所需的环境和配置进行打包，方便大规模部署
- **Why we use Docker?**
  - 与虚拟机相比，Docker更加轻量级

	Docker	虚拟机
启动	s	m
硬盘占用	M	G
性能	接近宿主	弱于宿主

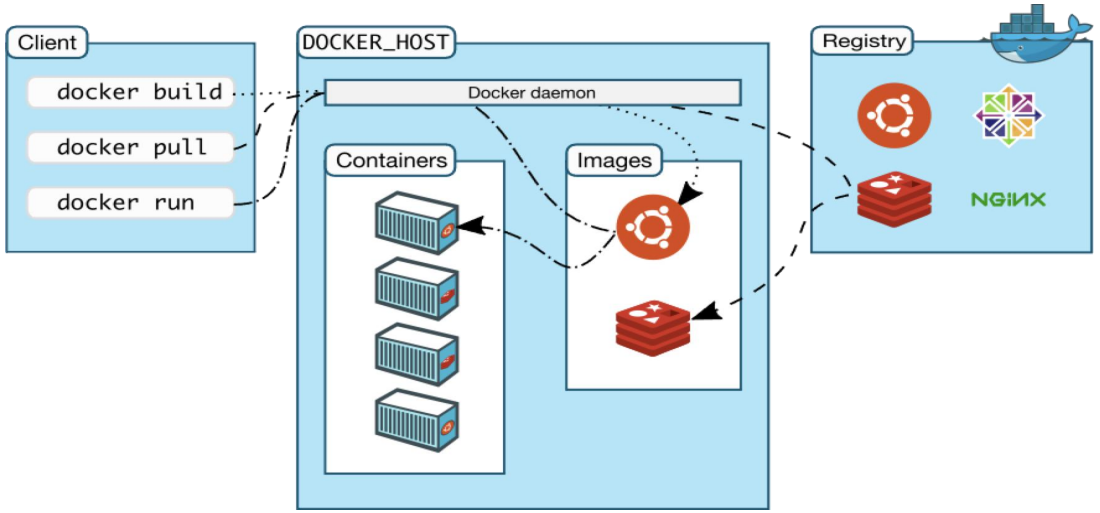
Docker基本概念

	Docker	虚拟机
启动	s	m
硬盘占用	M	G
性能	接近宿主	弱于宿主

- **Why we use Docker?**
  - 与虚拟机相比，Docker更加轻量级。
  - 使用docker可以有效避免仅仅提供源代码的情况下，程序运行所需的环境不同、配置不同等导致的兼容问题。从而使用docker可以轻松实现跨平台、跨服务器的运行，简化了从开发、调试到生成的迁移问题。
  - ~~只要开发过程中配置过环境的人都知道，光是配置环境半条命就没了~~
  - 部署小网站的时候我们要用到docker——本地我们写好并且能跑起来的东西，我们要通过docker将程序和配置环境打包起来，部署到云端的服务器上

Docker的基本概念

- **Docker的基本组成**
  - Docker主要由镜像（image）、容器（container）、仓库（repository）这三个部分组成
  - 一个类比：
    - 仓库是大家上传镜像、分享镜像的地方，相当于github。docker hub上面有一些重要的镜像，比如ubuntu等。
    - 镜像——类比与C++中的类模板，用于创造容器（container）
    - 容器——类比于C++中的对象，是大家写好的类模板（镜像）的实例化



Docker的基本概念

- 镜像（image）：利用docker将程序和程序运行所需的环境打包，得到的就是镜像。镜像本身包含了程序运行所需要的一切依赖。

  镜像是只读的，作为创建容器的模板存在。
- 容器（container）：镜像运行在容器中，每个容器都是一个独立化的实例。正如在C艹我们可以通过一个class类实例化得到很多对象一样，我们可用同一个镜像运行多个容器。

  容器和容器之间相互独立，互不干扰，保证运行的安全性。

目录

Docker常用命令

- docker的基本命令
  - docker build <image>：将源代码打包形成镜像并存放在本地
  - docker pull <image>：从docker hub上拉取镜像并存放在本地
  - docker run <image>: 运行镜像image，如果在本地找到了该镜像就直接启动；否则从docker hub上拉去image并启动
    - 参数-d：后台运行
    - 参数-p：端口映射
- 演示
  - **docker run hello-world**
  - **docker run ubuntu**
  - **docker run -it ubuntu:latest /bin/bash**

Docker常用命令

- **docker info**: 查看docker信息，如占用空间、版本等
- **docker --help**: 获取docker的帮助信息
- **docker <具体命令> --help** ： 获取docker <具体命令>更详细的信息
- **docker images**: 列出所有本地镜像&详细信息
  - 参数： -a ： 列出所有本地镜像，含历史映像层
  - 参数： -q ： 只显示镜像ID
- **docker search <image>**: 搜索远程仓库是否有相应的image

Docker常用命令

- **docker pull <image>[:tag]**: 拉取镜像
  - example: **docker pull centos:latest**
- **docker ps** : 查看运行中的**容器**
- **docker rmi <imageID or imageName> [options]**: 删除镜像
  - **-f: 如果image正在被停止的容器无法删除，这时加上-f这个参数就可以被强制删除**
  - **example: （删除全部）docker rmi -f $(docker images -qa)**
  - **支持同时删除多个**
- **docker rm [Options] <ID or Names>**: 删除容器
- **docker exec [OPTIONS] CONTAINER COMMAND**: 进入容器内部进行调试

Docker常用命令

- docker save: 将镜像打包成tar包
  - example: docker save demo > demo.tar
- docker load：从tar包加载一个镜像
  - example: docker load < backend.tar

Docker常用命令

- **docker export 容器ID > 文件名.tar**
  - 导出容器内部的内容作为tar归档文件（导出整个容器并进行备份）
- **docker import： 将export的容器恢复**

目录

使用Dockerfile创造镜像

- Dockerfile简介：Dockerfile是用来构建docker镜像的文本文件。是由一条条构建镜像所需的指令和参数构成的脚本。
- 使用Dockerfile创造镜像的大致流程：Dockerfile——>docker image ——> docker container
  1. 编写Dockerfile
  2. docker build ： 构建镜像
  3. docker run ： 以镜像为基础构建容器

一个Dockerfile的栗子

```dockerfile
# syntax=docker/dockerfile:1

FROM python:3.10-slim-buster

WORKDIR /app

COPY requirements.txt requirements.txt
RUN pip3 install -r requirements.txt

COPY . .

CMD ["python3", "-m" , "flask", "run", "--host=0.0.0.0"]

```

Dockerfile的一些规则
1. 文件名称为Dockerfile
2. # 表示注释
3. 每条保留字都必须是大写字母后面跟着一些参数
4. 每条指令从上倒下顺序执行

一个Dockerfile的栗子

```dockerfile
# syntax=docker/dockerfile:1

FROM python:3.10-slim-buster

WORKDIR /app

COPY requirements.txt requirements.txt
RUN pip3 install -r requirements.txt

COPY . .

CMD ["python3", "-m" , "flask", "run", "--host=0.0.0.0"]

```

具体的解释：

- # syntax=docker/dockerfile:1 指定syntax的版本为docker/dockerfile:1
- FROM 命令指定了我们所基于的镜像
- WORKDIR 命令指定了后续命令的工作目录
- COPY 命令将文件从外部复制到镜像中
- RUN 命令在构建镜像时运行
- COPY . . 将当前目录的所有文件复制到镜像的 /app 目录下
- CMD 指定了容器的启动命令，在 docker run 启动容器时运行

（演示环节）

Docker: 卷的挂载

- 命令格式：-v [host-dir]:[container-dir]:[rw|ro]
- example: docker run -it -v /e/demo:/src ubuntu

Docker: 卷的挂载

- **Q：卷的挂载有什么用？**
  - **方便编辑和方便共享**
    - 如果有一些数据想在多个容器间共享，或者想在一些临时性的容器中使用该数据，那么最好的方案就是你创建一个数据卷容器，然后从该临时性的容器中挂载该数据卷容器的数据。 这样，即使删除了刚开始的第一个数据卷容器或者中间层的数据卷容器，只要有其他容器使用数据卷，数据卷都不会被删除的。
  - **防止丢失**
    - docker rm my_container删除相应的容器时，挂载出来的数据不会丢失
    - 彻底删除使用docker rm -v my_container，避免不必要数据的存在
  - **备份方便**
    - **不能使用docker export、save、cp等命令来备份数据卷的内容**，因为数据卷是存在于镜像之外的。备份的方法可以是创建一个新容器，挂载数据卷容器，同时挂载一个本地目录，然后把远程数据卷容器的数据卷通过备份命令备份到映射的本地目录里面。

Docker: 卷的挂载

- **Q：具体点、？卷是神马？为神马要卷的挂载？**

https://blog.csdn.net/fragrant_no1/article/details/83184635

- Docker镜像是由多个文件系统（只读层）叠加而成。当我们启动一个容器的时候，Docker会加载只读镜像层并在其上（即镜像栈顶部）添加一个读写层。如果运行中的容器修改了现有的一个已经存在的文件，那该文件将会从读写层下面的只读层复制到读写层，该文件的只读版本仍然存在，只是已经被读写层中该文件的副本所隐藏。当删除Docker容器，并通过该镜像重新启动时，之前的更改将会丢失（因为重新创建了容器）。在Docker中，只读层及在顶部的读写层的组合被称为 **UnionFile System（联合文件系统）** 。
- Docker中的数据可以存储在类似于虚拟机磁盘的介质中，在Docker中称为数据卷（Data Volume）。数据卷可以用来存储Docker应用的数据，也可以用来在Docker容器间进行数据共享。数据卷呈现给Docker容器的形式就是一个目录，支持多个容器间共享，修改也不会影响镜像。使用Docker的数据卷，类似在系统中使用 `mount` 挂载一个文件系统。
- 为了能够保存（持久化）数据以及共享容器间的数据，Docker提出了Volume的概念。简单来说，Volume就是目录或者文件，它可以绕过默认的联合文件系统，而以正常的文件或者目录的形式存在于宿主机上。

谢谢大家！

张竞择

THU-Departement of Automation

jz-zhang21@mails.tsinghua.edu.cn

参考资料

[1] Dockerfile文档官方教程：
https://docs.docker.com/engine/reference/builder/
[2] Docker desktop安装：https://docs.docker.com/get-docker/
[3] Docker 命令行：
https://docs.docker.com/engine/reference/commandline/cli/
[4] 卷的挂载
https://blog.csdn.net/fragrant_no1/article/details/83184635