第二课：Docker

1. docker的安装及为什么要选择docker

为什么选择docker

启动速度快；环境一致；占用资源少；部署容易

wsl下的安装

安装docker desktop 并在安装时/设置里开启wsl支持。（可自行百度/deepseek）

配置镜像：

代码块

```
bash <(wget -qO- https://xuanyuan.cloud/docker.sh)
```

vm下的安装

代码块

```
# 更新系统并安装依赖
sudo apt-get update
sudo apt-get install ca-certificates curl

# 创建密钥目录
sudo install -m 0755 -d /etc/apt/keyrings

# 下载并添加 Docker GPG 密钥（使用中科大镜像）
sudo curl -fsSL https://mirrors.ustc.edu.cn/docker-ce/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# 添加 Docker 中科大镜像源
echo \
  "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc] \
  https://mirrors.ustc.edu.cn/docker-ce/linux/ubuntu \
  $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable"
| \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
18
19  # 更新索引
20  sudo apt-get update
21
22  # 安装docker
23  sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-
    plugin docker-compose-plugin
24
```

配置镜像：

```
代码块
1   bash <(wget -qO- https://xuanyuan.cloud/docker.sh)
```

实践：帮助安装docker

2. 镜像与容器操作

`docker pull` 拉取镜像

`docker images` 查看现有所有镜像

`docker ps` 查看现在正在运行的容器

`docker run` 运行容器（参数待补充）

`docker ps -a` 查看所有创建的容器（包括未运行的）

`docker stop` 停止某个容器

`docker start` 启动某个容器

`docker restart` 重启某个容器

`docker logs` 查看某个容器的日志

`docker exec -it /bin/bash` 进入容器内部并使用bash作为默认shell

Docker run的参数：-d 后台运行；-p 端口映射（8080:80）宿主机:容器；--name 指定容器名称；-e 设置环境变量；-it 交互模式+终端

实践：运行一个nginx

3. Dockerfile与镜像构建

`FROM` 指定基础镜像：如 `FROM ubuntu:22.04`

`RUN` 执行命令：如 `RUN apt-get update && apt-get install -y python3`

`COPY` 复制本地文件至镜像内：如 `COPY . /app` 左侧是相对于Dockerfile的本地路径，右侧是容器内路径

`WORKDIR` 设置工作目录：如 `WORKDIR /app` 这行命令后续的所有命令都会在此目录（/app）下运行

`CMD` 定义容器启动时默认运行的命令：如 `CMD ["python3", "app.py"]`

`ENTRYPOINT` 主入口命令：如 `ENTRYPOINT ["python3"]` CMD提供的命令可被外部传入变量覆盖，如：

代码块

```
1  ENTRYPOINT ["python3"]
2  CMD ["app.py"]
```

然后使用 `docker run myimage test.py` 实际上运行的是 `python3 test.py` 可以看到CMD提供的app.py被传入的test.py覆盖掉了。

`ENV` 设置环境变量，如某些秘钥需要通过环境变量传入程序：如 `ENV TZ=Asia/Shanghai`

`EXPOSE` 设置服务对docker内部的暴露端口号：如 `EXPOSE 80` 就是对docker服务内暴露了80端口，可以使用-p至本机端口

`docker build -t myimage .` 根据指定路径的Dockerfile构建docker镜像（目前程度了解到.即可，没必要往下了解）

实践：将我们提供的项目打包成镜像

4. 数据卷挂载及docker compose

`-v /host/path:/container/path` 冒号前为本地路径，冒号后为容器内路径

实践：将本地目录挂载搭配到nginx容器内，实现托管网页

`docker-compose.yml` 简单来说就是 `docker run` 的集合

`docker compose up -d` 启动compose的所有容器

`docker compose restart` 重启compose的所有容器

`docker compose down` 停止compose的所有容器，并删除容器、网络等

实践：启动一个compose