Docker Cheat Sheet

By GeekHour @ bilibili

镜像管理

docker image ls / docker images	查看镜像
docker search [image] eg. docker search nginx	检索镜像
docker pull [image]	拉取镜像
docker push [image] eg. docker push geekhour/hello-docker:latest	上传镜像
docker save [image] -o FILE / docker save [image] > FILE eg. docker save geekhour/hello-docker:latest > hello-docker.tar	保存镜像
docker load -i FILE eg. docker load -i hello-docker.tar	导入镜像
docker history [image]	查看镜像历史
docker rmi [image] / docker image rm [image]	删除镜像
docker image prune	删除不再使用的镜像
docker import [URL/FILE]	将文件系统导入为镜像
docker commit [container] [image]	从容器创建镜像

网络管理

docker network ls	列出可用网络
docker network inspect [network]	查看网络详细信息
docker network create [network]	创建一个新的网络
docker network rm [network]	删除一个网络
docker network connect [network] [container]	将容器连接到网络
docker network disconnect [network] [container]	将容器从网络断开

插件管理

docker plugin ls	列出插件
docker plugin install [plugin]	安装插件
docker plugin enable [plugin]	启用插件
docker plugin disable [plugin]	禁用插件
docker plugin rm [plugin]	卸载插件

容器管理

docker create [image]	创建容器（仅创建，不运行）
docker run [image]	创建并运行容器
docker start [container]	启动容器
docker stop [container]	停止容器
docker restart [container]	重启容器
docker ps / docker container ls	列出正在运行的容器
docker ps -a / docker container ls -a	列出所有容器
docker exec -it [container] bash / docker attach [container]	以交互模式进入容器
docker export [container] -o FILE / docker export [container] > FILE	导出容器
docker import FILE	导入容器快照
docker logs [container]	查看容器日志
docker rm [container] / docker container rm [container]	删除容器
docker port [container]	查看容器端口映射
docker top [container]	显示容器内进程
docker cp [FILE] [container]:[PATH]	复制本地文件 到容器内的指定路径
docker diff [container]	显示容器内的变化
docker stats [container]	显示容器资源使用情况

数据卷管理

docker volume create [volume]	创建一个数据卷
docker volume ls	查看数据卷
docker volume inspect [volume]	查看数据卷详细信息
docker volume rm [volume]	删除数据卷
docker volume prune	删除所有未使用的数据卷

日常操作

docker info	查看 docker 系统信息
docker version	查看 Docker 版本
docker --help	查看 Docker 帮助文档
docker [command] --help	查看 Docker 命令帮助
docker login/logout	登录/退出 DockerHub

容器运行

语法格式：
docker run [options] image [command] [arg...]

docker run --name [name] [image]
创建 运行并命名容器

docker run -d [image]
创建一个容器并后台运行

docker run -p [hostPort]:[containerPort] [image]
创建一个容器并指定端口映射

docker run -P [image]
创建一个容器并指定端口映射（随机分配）

docker run -e [key=value] [image]
创建一个容器并指定环境变量

docker run -w [PATH] [image]
创建一个容器并指定工作目录

docker run -name [name] [image]
创建一个容器并指定容器名称

docker run [image] [command]
创建一个容器并在容器中执行命令（交互模式）

docker run -d -p [hostPort]:[containerPort] -e [key=value]
-w [PATH] --name [name] [image]
创建一个容器，并指定容器名称，后台运行，端口映射
环境变量，工作目录

eg. docker run -it nginx:latest /bin/bash
使用镜像 nginx:latest 来启动一个容器，
并在容器内执行交互式 bash shell

eg. docker run -it -p 3316:3306 -v /data:/data -d mysql:latest
创建一个 mysql 容器，后台模式启动，主机 80 端口
映射到容器 80 端口，主机 /data 目录映射到容器 /data 目录

常用 Dockerfile 指令

FROM [base_image]
指定基础镜像，必须为 Dockerfile 的第一条指令；

ADD
用于将文件复制到镜像中，源可以使 URL 或者本地文件，也可以一个压缩文件（自动解压）；

COPY [--chown=<user>:<group>] [源路径] [目标路径]
用于将文件拷贝到镜像中，源只能是本地文件；

WORKDIR [PATH]
用于指定工作目录，可以使用多个 WORKDIR 指令，如果使用相对路径，则是相对于上一条 WORKDIR 指令所指定的目录；

ENV <key> <value> /
ENV <key1>=<value1> <key2>=<value2> ...
用于设置环境变量

CMD <命令> /
CMD ["可执行文件", "参数 1", "参数 2" ...]
用于指定默认的容器主进程，每个 Dockerfile 中只能有一条 CMD 指令，如果有多条，则只有最后一条会生效；

VOLUME <路径> /
VOLUME ["路径 1", "路径 2"...]
用于定义匿名卷（持久化目录）