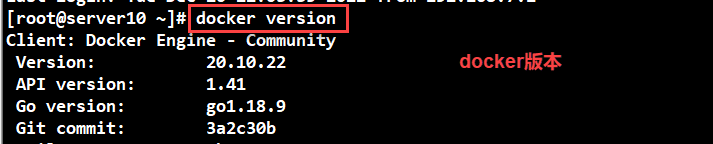
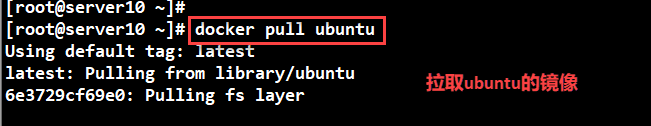
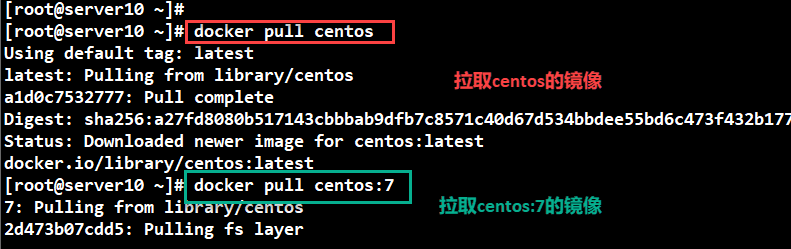
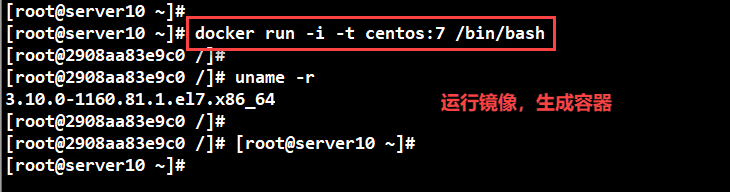
# Docker构建镜像，拉取镜像，生成容器以及创建仓库



**1.拉取docker 镜像**



**2.运行镜像，生成容器**



ctrl+p ctrl+q 返回到主机，容器还在运行

exit 退出容器，容器停止运行

**容器启动**

**docker run [OPTIONS] IMAGE [COMMAND] [ARG...]**

**-d : 后台运行容器，并返回容器ID**

**-i：以交互模式运行容器，通常与 -t 同时使用**

**-t：为容器重新分配一个伪输入终端，通常与 -i 同时使用**

**-v：绑定挂载目录**

**--name="mycontainer": 为容器指定一个名称**

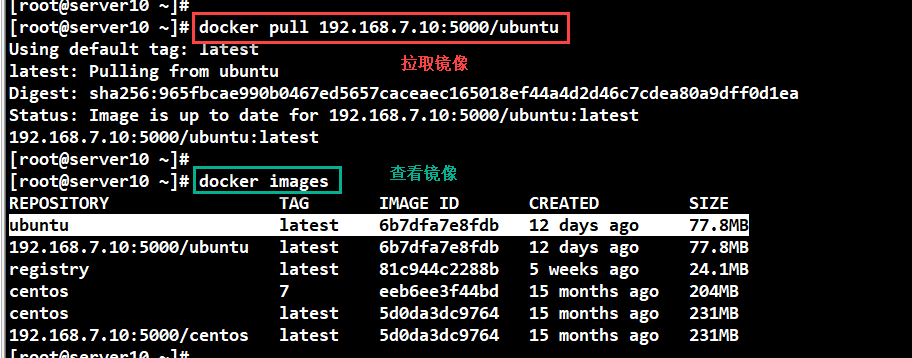
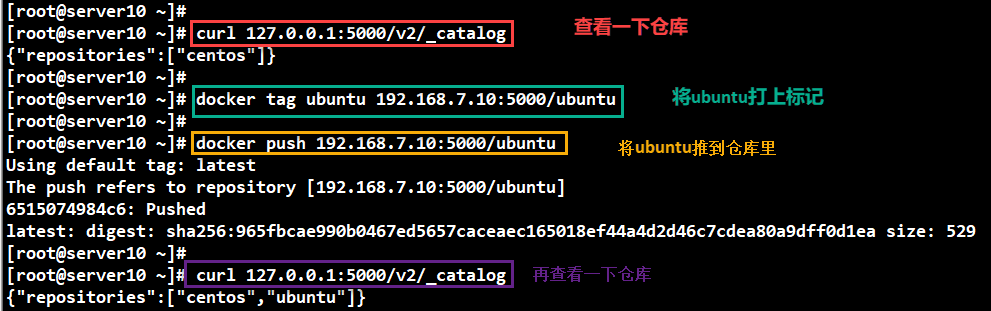
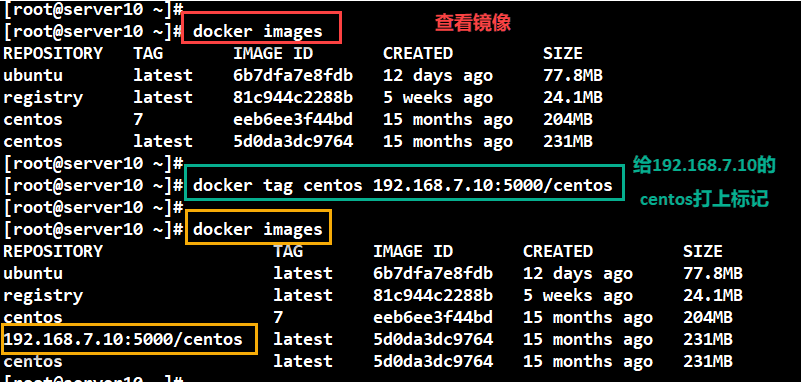
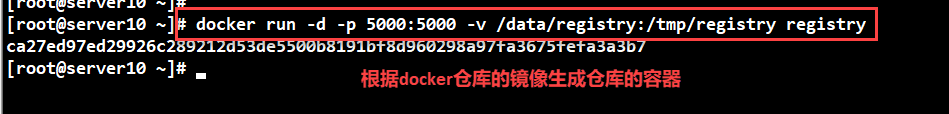
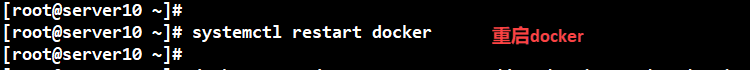
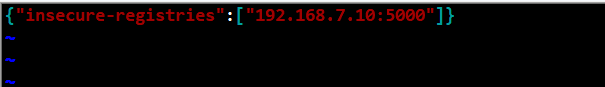
**--net="bridge": 指定容器的网络连接类型，支持如下：**

**bridge / host / none / container:<name|id>**

**-p/-P :端口映射**

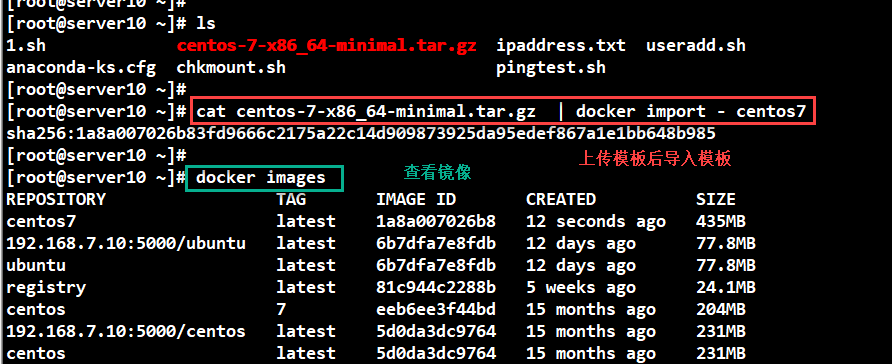


**3.构建私有仓库**

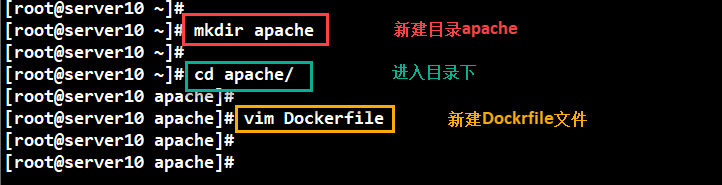


**4.Docker镜像的创建方法：**

**1）基于本地模板创建**



**2）基于Dockerfile文件创建**



**Dockerfile的基本结构**

**Dockerfile 一般分为四部分：基础镜像信息、维护者信息、镜像操作指令和容器启动时执行指令，’#’ 为 Dockerfile 中的注释。**

**FROM：指定基础镜像，必须为第一个命令**

**MAINTAINER: 维护者信息**

**RUN：构建镜像时执行的命令**

**ADD：将本地文件添加到容器中，tar类型文件会自动解压(网络压缩资源不会被解压)，可以访问网络资源，类似wget**

**COPY：功能类似ADD，但是是不会自动解压文件，也不能访问网络资源**

**CMD：构建容器后调用，也就是在容器启动时才进行调用。**

**ENTRYPOINT：配置容器，使其可执行化。配合CMD可省去"application"，只使用参数。**

**LABEL：用于为镜像添加元数据**

**ENV：设置环境变量**

**EXPOSE：指定于外界交互的端口**

**VOLUME：用于指定持久化目录**

**WORKDIR：工作目录，类似于cd命令**

**USER:指定运行容器时的用户名或 UID**

**ARG：用于指定传递给构建运行时的变量**

**ONBUILD：用于设置镜像触发器**

