**[Docker学习笔记之二，基于Dockerfile搭建JAVA Tomcat运行环境](http://blog.csdn.net/junjun16818/article/details/34845791)**

**前言**

在第一篇文字中，我们完全人工方式，一个命令一个命令输入，实现一个java tomcat运行环境，虽然也初见成效，但很累人。如果依靠依靠脚本构建一个Tomcat容器实例，一个命令可以搞定，何乐而不为呢。好在Docker提供了Dockerfile作为构建Docker镜像脚本，避免人们一行一行的输入，真是善莫大焉。Dockerfile脚本可以做到随时维护修改，即可以分享，更有利于在模板化，更不用说传输了，好处那是一大箩筐！

最终目的：打造一个支持SSH终端登录、Tomcat7自动运行的Docker镜像。

**编写一个Dockerfile**

所有环境，同第一篇文字。在ubuntu下安装好vim之后（吐槽一下，ubuntu系统内置vi命令十分难用，只好求助于vim )：

sudo vim Dockerfile

编辑一个Dockerfile文件了，下面就是具体文件内容：

# VERSION 0.0.1

# 默认ubuntu server长期支持版本，当前是12.04

FROM ubuntu

# 签名啦

MAINTAINER yongboy "yongboy@gmail.com"

# 更新源，安装ssh server

RUN echo "deb http://archive.ubuntu.com/ubuntu precise main universe"> /etc/apt/sources.list

RUN apt-get update

RUN apt-get install -y openssh-server

RUN mkdir -p /var/run/sshd

# 设置root ssh远程登录密码为123456

RUN echo "root:123456" | chpasswd

# 添加orache java7源，一次性安装vim，wget，curl，java7，tomcat7等必备软件

RUN apt-get install python-software-properties

RUN add-apt-repository ppa:webupd8team/java

RUN apt-get update

RUN apt-get install -y vim wget curl oracle-java7-installer tomcat7

# 设置JAVA\_HOME环境变量

RUN update-alternatives --display java

RUN echo "JAVA\_HOME=/usr/lib/jvm/java-7-oracle">> /etc/environment

RUN echo "JAVA\_HOME=/usr/lib/jvm/java-7-oracle">> /etc/default/tomcat7

# 容器需要开放SSH 22端口

EXPOSE 22

# 容器需要开放Tomcat 8080端口

EXPOSE 8080

# 设置Tomcat7初始化运行，SSH终端服务器作为后台运行

ENTRYPOINT service tomcat7 start && /usr/sbin/sshd -D

需要注意：

1. ENTRYPOINT，表示镜像在初始化时需要执行的命令，不可被重写覆盖，需谨记
2. CMD，表示镜像运行默认参数，可被重写覆盖
3. ENTRYPOINT/CMD都只能在文件中存在一次，并且最后一个生效 多个存在，只有最后一个生效，其它无效！
4. 需要初始化运行多个命令，彼此之间可以使用 *&&* 隔开，但最后一个须要为无限运行的命令，需切记！

ENTRYPOINT/CMD，一般两者可以配合使用，比如：

ENTRYPOINT ["/usr/sbin/sshd"]

CMD ["-D"]

在Docker　daemon模式下，无论你是使用ENTRYPOINT，还是CMD，最后的命令，一定要是当前进程需要一直运行的，才能够防容器退出。

以下无效方式：

ENTRYPOINT service tomcat7 start #运行几秒钟之后，容器就会退出

CMD service tomcat7 start #运行几秒钟之后，容器就会退出

这样有效：

ENTRYPOINT service tomcat7 start && tail -f /var/lib/tomcat7/logs/catalina.out

# 或者

CMD service tomcat7 start && tail -f /var/lib/tomcat7/logs/catalina.out

这样也有效：

ENTRYPOINT ["/usr/sbin/sshd"]

CMD ["-D"]

具体请参考官方文档：Dockerfiles for Images

**构建镜像**

脚本写好了，需要转换成镜像：

docker build -t yongboy/java7 .

**-t**： 为构建的镜像制定一个标签，便于记忆/索引等

**.** ： 指定Dockerfile文件在当前目录下

网速不太好，会等待很长时间。很多操作可能需要科学上网，逼得我只能一直挂着VPN，方能畅通无阻。

构建镜像完成之后，看看运行效果：

docker run -d -p 22 -p 8080:8080 yongboy/java7

在运行命令中，还得需要显式指定 -p 22 -p 8080:8080，否则在Docker 0.8.1版本中不会主动映射到宿主机上。据悉在Docker 0.4.8版本时，就不担心这个问题。 或者，您要有好的方式，不妨告知于我，谢谢。

在Dockerfile中，若没有使用ENTRYPOINT/CMD指令，若运行多个命令，可以这样做：

docker run -d -p 22 -p 8080 yongboy/java7 /bin/sh -c "service tomcat7 start && /usr/sbin/sshd -D"

**提交/保存镜像**

创建好的镜像，可以保存到索引仓库中，便于下次使用（当然，我们直接共享Dockerfile，是最简单的事情，:)) ），但毕竟镜像可以做到开箱即用。

1. https://index.docker.io/ 注册一个账号，例如yongboy
2. 构建镜像
3. docker build -t yongboy/java7 .

上面已经构建OK的话，可省略此步。

1. 登陆
2. docker login
3. 提交到Docker索引仓库
4. docker push yongboy/java7

现在可以起来喝杯热水，出去溜达会，也不一定能够上传完毕，那叫一个慢啊！

上传OK的话，可以得到类似地址：https://index.docker.io/u/yongboy/java7/

1. 如何使用镜像
2. docker pull yongboy/java7

剩下的步骤，就很简单了。

**小结**

有关Dockerfile的进阶阅读：

* <http://www.docker.io/learn/dockerfile/level2/>
* <http://www.docker.io/learn/dockerfile/level2/>