链滴

我的第一个 Dockerfile: jdk8 + maven3.6

作者：CodingOX
原文链接：https://ld246.com/article/1555668677973
来源网站：链滴

前言

最近在学习Docker，在学习Dockerfile的时候，感觉有些吃力，所以就停了下来，一边参考大神们的码，一边改，一边写，这个过程中沉淀了一些东西，这里记录下，也算做个分享。

学习资料

Docker的书网上好多，我阅读了好几本，发现好多都是抄袭《第一本Docker书 修订版》，这个在github上有，这里给一个链接吧：《第一本Docker书 修订版》，如果你是初学者，建议看这本书。

同时再推荐个掘金上的小册：开发者必备的 Docker 实践指南。

当然DockerHub上的各个镜像也是不错的，其中有很多都带有Dockerfile，可以根据他们进行阅读。

第一个脚本

说明

这个脚本是我为了使用最新的JDK8_201版本和maven3.6.1构建的。

如果你的JDK8是商用，最新的版本都得是202，否则会被Oracle认为你侵权。

在构建构成中，我也发现了一些问题，主要如下：

1. 通过 wget下载oracle jdk很恶心，官方只给你最新的版本的下载，如果老版本，你必须要登录。
2. 通过 wget下载的速度，真心慢，这个应该是国情吧...
3. 为了处理上述问题，我是先下载到本地后，通过 COPY引入的，这样的好处是构建镜像的速度快缺点是没有直接利用远程下载来的方便。

准备工作

为了使用这个脚本，你需要先创建一个目录，比如是dockerfile-helloworld，同时下载3个东西放在个目录中。

1. jdk-8u201-linux-x64.tar.gz ，下载地址：https://www.oracle.com/technetwork/java/javase/downloads/java-archive-javase8-2177648.html
2. jce_policy-8.zip，下载地址：http://download.oracle.com/otn-pub/java/jce/8/jce_policy-8.zip
3. apache-maven-3.6.1-bin.tar.gz，下载地址：http://archive.apache.org/dist/maven/maven-3/.6.1/binaries/apache-maven-3.6.1-bin.tar.gz

编写脚本

```
FROM jeanblanchard/alpine-glibc
MAINTAINER "liuchunfu" <liuchunfu@outlook.com>

ENV M2_HOME=/usr/mvn
ENV JAVA_HOME=/usr/java
ENV PATH=${PATH}:${JAVA_HOME}/bin:${M2_HOME}/bin
```

```
# 因为Oracle的JDK不能下载，所以先下载在本地,什么都不带，默认应该是放在temp目录下下的
COPY jdk-8u201-linux-x64.tar.gz /temp/java.tar.gz
RUN  mkdir -p ${JAVA_HOME} && \
  tar -zxf /temp/java.tar.gz -C  /temp && \
  mv /temp/jdk1.8.0_201/*  ${JAVA_HOME}

# 处理jdk对于AES256不能加密的问题
COPY jce_policy-8.zip /temp/jce_policy-8.zip
RUN  unzip /temp/jce_policy-8.zip -d /temp && \
   mv /temp/UnlimitedJCEPolicyJDK8/*.jar ${JAVA_HOME}/jre/lib/security/

# 删除jdk目录下一些可能用不到的资源
RUN rm -rf ${JAVA_HOME}/*src.zip \
        ${JAVA_HOME}/lib/missioncontrol \
        ${JAVA_HOME}/lib/visualvm \
        ${JAVA_HOME}/lib/*javafx* \
        ${JAVA_HOME}/jre/lib/plugin.jar \
        ${JAVA_HOME}/jre/lib/ext/jfxrt.jar \
        ${JAVA_HOME}/jre/bin/javaws \
        ${JAVA_HOME}/jre/lib/javaws.jar \
        ${JAVA_HOME}/jre/lib/desktop \
        ${JAVA_HOME}/jre/plugin \
        ${JAVA_HOME}/jre/lib/deploy* \
        ${JAVA_HOME}/jre/lib/*javafx* \
        ${JAVA_HOME}/jre/lib/*jfx* \
        ${JAVA_HOME}/jre/lib/amd64/libdecora_sse.so \
        ${JAVA_HOME}/jre/lib/amd64/libprism_*.so \
        ${JAVA_HOME}/jre/lib/amd64/libfxplugins.so \
        ${JAVA_HOME}/jre/lib/amd64/libglass.so \
        ${JAVA_HOME}/jre/lib/amd64/libgstreamer-lite.so \
        ${JAVA_HOME}/jre/lib/amd64/libjavafx*.so \
        ${JAVA_HOME}/jre/lib/amd64/libjfx*.so

#处理Maven
COPY apache-maven-3.6.1-bin.tar.gz  /temp/maven.tar.gz
RUN mkdir $M2_HOME && \
  tar -zxf /temp/maven.tar.gz -C /temp && \
  mv /temp/apache-maven-3.6.1/*  $M2_HOME

#直接把temp这个临时目录删除，减少镜像的地址
RUN rm -fr /temp
```

在上述脚本中使用了如下几个指令：

- Run 这个网上解释很多，我觉的你可以想象为，你进入一个linux系统后，要部署java环境，你要通终端输入哪些指令。
- COPY就是将资源放入镜像内，为什么要放入其中了，因为是为了使用他(这里我感觉貌似可以直接部引用)

核心的其实就这2个指令，其他的都是linux的脚本。

慢慢发现了一个问题，我觉的编写Dockerfile不难，难得是shell脚本的编写啊...

原文链接：我的第一个 Dockerfile: jdk8 + maven3.6

结语

最后，我把这个东西开源了，放在了github上，地址:来点我啊来啊————，今年第一次开源的小西，好了周五6点10分了，下班~

原文链接：我的第一个 Dockerfile: jdk8 + maven3.6