链滴

docker-compose 一键部署 Nginx+Tomcat+Mysql+solo2.7.0

作者：liumapp

原文链接：https://ld246.com/article/1527837772238

来源网站：链滴

服务器环境的搭建一直是一个对新手不友好的环节，但借助容器技术Docker和编排工具Docker-Compose，我们可以非常方便的一次性完成配套环境的搭建及项目发布。

前言

先上项目代码： liumapp/solo-in-docker

拷贝到本地目录后，先运行build-image.sh脚本安装相关Docker镜像：Tomcat9, Mysql5以及Ngin
。

至于solo，则是直接在本地打包编译好，然后放置在webapps目录下，通过/conf/server.xml的配置来让Tomcat进行加载。首次运行的伙伴们可以不作修改直接使用。

整体环境配置

对于环境配置，我会做拆分来讲，因为单个容器的环境和多个容器的环境在运行上是有区别的，容器容器之间的连接也要考虑。

打个比方，在Nginx容器中的localhost:8080并不会指向Tomcat容器中所监听的8080端口。

整体环境的配置，如果一个一个Dockerfile去写，那么是相当麻烦的，好在Docker有一个名为Docker Compose的工具提供，我们可以使用它一次性完成整体环境的配置：

首先我们看看docker-compose.yml配置文件的内容：

```
version: "3"

services:
  mysql:
    container_name: mysql
    image: mysql:5.5.60
    restart: always
    volumes:
      - ./mysql/data:/var/lib/mysql
      - ./mysql/conf/mysqld.conf:/etc/mysql/mysql.conf.d/mysqld.cnf
    ports:
      - "6603:3306"
    environment:
      - MYSQL_ROOT_PASSWORD=adminadmin
  nginx:
    container_name: nginx
    restart: always
    image: nginx:1.13
    ports:
      - 80:80
      - 443:443
      - 5050:5050
      - 4040:4040
    volumes:
      - ./conf/vhosts:/etc/nginx/conf.d
      - ./logs:/var/log/nginx
      - ./www/:/var/www/
```

```
    links:
      - tomcat:t1
  tomcat:
    container_name: tomcat
    restart: always
    image: tomcat:9.0
    ports:
      - 8080:8080
      - 8009:8009
    volumes:
      - ./webapps/solo.war:/usr/local/tomcat/webapps/solo.war
      - ./conf/server.xml:/usr/local/tomcat/conf/server.xml
      - ./logs:/usr/local/tomcat/logs
    links:
      - mysql:m1
```

可以看到，我们一共设置了三个service，分别是mysql, nginx, tomcat，其中，需要注意的地方是它的volumes以及links，其他的地方相信大家很容易理解。

mysql环境配置

首先看看最简单的mysql，它没有设置links，因为是其他容器连接它，它也就不需要理会别的家伙了但是mysql的volumes却是最为重要的，如果我们在这里不设置volumes的话，那么很棒，每一次docer重启，或者mysql的container重启，您辛辛苦苦产生的database数据就会一夜回到解放前：“啥没有”。

所以mysql的volumes，就设置了两点：

- 指定mysql产生的data文件需要同步到宿主机的./mysql/data目录下
- 指定mysql的配置文件从宿主机的./mysql/conf/mysqld.conf读取

那么至于mysql的配置文件中定义了什么内容，就由伙伴们自行研究吧。

nginx环境配置

接下来我们看看Nginx，它有volumes，也有links，那么这个地方的volumes我们就不解释了，大家以参照mysql的解释来理解，重点看看links，我们可以发现，Nginx的links建立了与tomcat容器的连，为什么呢，配置过nginx+tomcat的伙伴们应该会知道，nginx负责监听80端口，tomcat负责监听880端口，nginx接收到需要由tomcat来处理的请求就将请求转发到8080。

那么在docker的环境下，nginx直接将请求转发到8080，tomcat会接受到吗？答案肯定是不能的。

我们必须先定义一个links，这里的值为t1，那么在nginx负责转发的配置文件里，就要写上：

```
upstream backend {
    server t1:8080;
}

server {
    listen       80;
    server_name  localhost
```

```
    access_log off;

    location / {
        proxy_pass http://backend$request_uri;
        proxy_set_header  Host $host:$server_port;
        proxy_set_header  X-Real-IP  $remote_addr;
        client_max_body_size  10m;
    }
}
```

来告诉nginx，tomcat在哪里。

tomcat环境配置

最后便是Tomcat了，它比较特殊，因为还有solo的war包需要发布上去。

但发布solo的war包我们只需要定义几个volumes的值便可以了。

```
volumes:
  - ./webapps/solo.war:/usr/local/tomcat/webapps/solo.war
  - ./conf/server.xml:/usr/local/tomcat/conf/server.xml
```

当然，server.xml的配置必不可少，但是这个我们也不多说了。